Abstract—Documentation of coral reefs and fish is usually performed in shallow waters by divers who carry small and inexpensive handheld video and still cameras. These cameras do not allow accurate measurements of distances due to the lack of stereo capacity or small camera baselines. This research developed a set of software modules that perform calibration and synchronization of a stereo-pair by using either a flashing light or image motion consistency. This allows a diver to perform stereo-vision measurements by means of two low-cost video cameras operating independently. We assume that the intrinsic calibration and the distance between the cameras are known, while the precise camera orientation is computed by the system without using a calibration grid.

I. INTRODUCTION

People studying coral reefs and fish require methods and setups for the documentation of these marine biodiversity. The studies are usually performed in shallow waters by divers who carry handheld video and still cameras. Research has been conducted on determining the accuracy and precision of stereo-video measurements of moving fish as an accurate indicator of their true length [3]. It states that an additional criterion for stereo-video measurement was that the fish had to be at an angle of less than approximately 50° perpendicular to the stereo-video cameras to ensure maximum accuracy. The results of the research show that stereo-video can be used to make accurate field measurements of the length of reef fish and has many advantages in monitoring programs that aim to detect changes in the mean length of fish that are rare, or are sampled in low densities [3].

When capturing the video of coral reefs and fishes using multiple cameras, the cameras are usually attached to a vehicle or handheld by divers. Sometimes, the cameras are not securely positioned or not rigidly attached. Especially in low-cost imaging systems using off-the-shelf video cameras, the resulting videos may suffer from the cameras’ slight changes in orientation due to water resistance while moving. Also, most commercially available underwater stereo-systems have very small camera baselines allowing them to be more compact and rugged. Small baselines have a very negative impact on the ability to perform accurate triangulation, and thus provide useful distance measurements. This is even worse when acquiring images of marine animals and structures at larger distances. For these reasons, it is relevant to develop and validate stereo-imaging systems based on affordable cameras that do not rely on synchronized acquisition which requires cable link between the two cameras. Nonetheless, the use of unsynchronized and non-rigid stereo is a challenging problem because it requires the development of procedures for quick synchronization and calibration in the field.

II. RELATED WORKS

F. Oleari et al [5] made a low-cost stereo-vision system for object recognition. It uses two unsynchronized Logitech C270 UVC cameras put inside an aluminum rigid bar case to ensure that it can hold against vibrations and collisions. The system applies Fast Point Feature Histogram (FPFH) descriptors to acquire point clouds in order to accurately perform object recognition. The intrinsic and extrinsic calibration, the disparity image and the point cloud computation are done with ROS packages and libraries (stereo_image_proc). Although the author states the cameras are unsynchronized, no automatic synchronization is presented in this work.

P. Cerri et al [6] presents a new method for extrinsic parameters computation by using 10 computer-generated or real images of a ball rolling on a flat plane in front of the camera for computing roll and pitch angles. The calibration is achieved by an iterative Inverse Perspective Mapping (IPM) process that uses estimation of the ball gradient invariant as a stop condition. The method is suited to be used to quickly calibrate vision systems where a grid is not available. Although this could be an alternative for an extrinsic camera calibration, this method cannot be applied as it is because the research is aimed for underwater use and the calibration with a rolling a bowling ball underwater would be hard to implement.

Heng et al [7] made an automated pipeline that handles both intrinsic and extrinsic calibration of a stereo rig. It does not assume that there are overlapping fields of view. At first, for each generic camera, an automatic intrinsic calibration that requires a chessboard is conducted. Then, the vehicle must be driven around for a short time to perform the extrinsic calibration, i.e. find all camera-odometry transforms. Triangulation of the inlier feature point correspondences is generated by monocular visual odometry using the initial camera-odometry transforms and odometry data. The resulting sparse map is then optimized via bundle adjustment; the odometry poses are kept fixed while all 3D scene points and the camera-odometry transforms are optimized. In this work, we use the same way of performing the intrinsic calibration using a chessboard. However, instead of using the odometry in 3D, we apply a simpler approach of relating the two cameras using rotations along the 3D axis.

Dang et al [9] presents a framework for continuous stereo self-calibration. The active camera platform consists of three independently moving cameras: two stereo cameras with one rotational DOF (degree of freedom) and a telecamera with two DOF. The framework utilizes the geometry of stereo image sequences, stereo-image pairs and image
triplets. However our aim is to use a much simpler setup of unsynchronized cameras.

III. METHODOLOGY

A. Standard Calibration

First, a simple and quick intrinsic camera calibration using Camera Calibration Toolbox for Matlab [10] is conducted. This calibration technique makes use of a regularly gridded checkered pattern where the metric size of the rectangulars is known. This grid is flat, and therefore multiple images of the calibration grid are acquired at different distances and orientations. The images are then automatically corrected for the radial distortion by the toolbox. These corrected images are used to do a stereo calibration between the left and the right camera using the toolbox. In the stereo calibration, all intrinsic and extrinsic parameters are recomputed together with all the uncertainties so as to minimize the reprojection errors on both cameras for all calibration grid locations. The purpose of the standard calibration is two-fold. Firstly it allows to determine the intrinsic calibration for the two cameras. We assume that these parameters are not changing over time. Secondly, one of the objectives of this work is to avoid having to do the extrinsic calibration in the field each time that the stereo system geometry is changed. However, the standard stereo calibration is performed for comparison purposes.

B. Synchronization With a Flashing Light

When recording the same scene by two unsynchronized cameras, there will be a slight time inconsistency between them. This is because their recording start times are not exactly the same. To synchronize the cameras, the frame difference of the resulting video sequences must be determined. One of the simplest ways to do this is to capture a simultaneous event that can easily be recognized in both video sequences, such as a flashing light.

When both cameras are recording, a strong light is turned on for an instant. Therefore, in both videos, there will be a frame with a sudden impact peak compared with the averages of the previous and next frames. To detect the peak, the image is first converted into a gray scale which helps to simplify the thresholding of the white values. The criterion to detect the presence of the light is by counting the number of pixels that are present in the largest connected group after applying a threshold to the gray scale image. The number of pixels in the region for each connected component is measured and the connected component that have an approximation of the 3D rotation matrix that relates each image and the next one. The transformation matrix is directly computed using the coordinates of the markers fixed on a moving body. This can be done if more than 3 markers are fixed on the body. Then, the orientation angles and the center of rotation can be computed from all of the inliers. From H, an approximation to a rotation matrix is obtained by \( \tilde{R} = K^{-1}HK \). However, this approximation is not a good model for the inliers. Pure rotations avoid parallax effects and are easier to estimate robustly using 2D registration methods. Although it is impossible to execute pure rotations with a stereo rig due to the baseline distance, these movements can be executed while pointing the cameras to a faraway location, thus reducing the parallax effect.

D. Estimation of rotation velocities

From the acquired video, the extracted image sequences are corrected for the radial distortion. For each sequence, the images are matched sequentially over time using SIFT features and RANSAC for outlier removal. For each image, matched points are sought within the current and the next image.

The robust estimation algorithm uses homography of 8 DOF as the motion model to identify the inliers. The 8 DOF homography is the most general model that can be computed linearly and can describe a pure rotation of the camera in all three axes \((\alpha, \beta, \gamma)\). For RANSAC, a linear model is important since it can be computed very quickly and therefore the robust estimation can identify the outliers in less than one second after doing approximately 500 iterations of random sampling over a set of more than 400 matched points. After RANSAC, a final 8DOF homography H is computed from inliers.

After finding the matched points, a rotation-only homography with 8 DOF is computed. This is done in order to have an approximation of the 3D rotation matrix that relates each image and the next one. The transformation matrix is directly computed using the coordinates of the markers fixed on a moving body. This can be done if more than 3 markers are fixed on the body. Then, the orientation angles and the center of rotation can be computed from all of the inliers. From H, an approximation to a rotation matrix is obtained by \( \tilde{R} = K^{-1}HK \). However, this approximation is not a real rotation matrix because it is not necessarily orthogonal, due to the noise in the matched points and departures from the assumed rotation only camera motion. Furthermore, its determinant is not necessarily 1 since it is defined up to scale. Therefore, a proper rotation matrix \( \hat{R} \) is estimated from \( \tilde{R} \) by means of the singular value decomposition (SVD). Let \( U, S, V^T \) be the SVD of \( \tilde{R} \) such that \( \tilde{R} = U \cdot S \cdot V^T \). Then a rotation matrix approximation to \( \hat{R} \) is [17]

\[
\hat{R} = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^T
\]

From \( \hat{R} \) a vector of rotation angles \((\alpha, \beta, \gamma)\) is extracted using the X-Y-Z fixed angle convention of

\[
d_t = [t]_X = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}
\]
To improve the results of the estimation of these angles a non-linear minimization was implemented in the following form:

$$\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma} = \arg \min \sum_{i=1}^{N} d^2(x_{t}^{i} + 1, K R_{3D}(\alpha, \beta, \gamma) K^{-1} \cdot x_{t}^{i})$$

Where N is the number of inliers, $x_{t}^{i}$ and $x_{t}^{i+1}$ are matched points in the frame t and the next frame $t+1$ and $d^2(\cdot, \cdot)$ is the squared distance between point projections in one image. The minimization was done with the Levenberg-Marquardt algorithm [16] available in Matlab. However, this additional step did not make any difference with respect to the results of the SVD approximation and was not used in practice.

Apart from the 3 angles $\alpha, \beta$ and $\gamma$, we used a forth one, $\theta$, corresponding to the total angle of rotation obtained from the axis-angle representation:

$$\theta = \arccos \left( \frac{\text{trace}(K) - 1}{2} \right)$$

The objective of having this additional angle was to check if it would be sufficient in practical applications to do the synchronization with just one angle instead of the three components $(\alpha, \beta, \gamma)$.

E. Determining the frame shift

Since the left and right cameras are not started at the same time, there will be a time difference for the rotation events. Although the rotation angle transformation between the two cameras is similar, there are still slight differences, because they are placed at opposite ends of the bar.

After angular velocities are estimated from the temporal matching along the image sequence, the noise in the signal is reduced by smoothing the angular velocity profiles using a Savitzky-Golay filter. From the smoothed signals, both left and right cameras measured cross correlation is computed and the peaks for all angles are localized. These peaks can be obtained by taking only a part of the whole signal where high angular velocities were recorded. This is because the time interval corresponding to two or three saccadic movements of the camera is usually sufficient to find a clear peak.

The frame difference estimated from the angular velocities of $\alpha, \beta, \gamma$, and $\theta$ may vary slightly. We consider the frame difference to be the mean of all four rounded to the nearest integer. In this work, we are not considering that the resolution of the frame lag between the left and right sequences can be below one frame interval. Although potentially this could be done by estimating the fractional locations of the peak using a polynomial fit.

F. Stereo extrinsic calibration from image registration

The required parameters are a vector that describes the baseline with respect to the reference frame of the left camera $T = [t_x, t_y, t_z]^T$, and a rotation $R_2$ of the reference frame of the right camera, also described in the left camera frame. The approach used is based on the concepts of structure from motion [14] and visual odometry[15] where the motion of a camera in 3D can be found from point matches.

The method uses the Essential matrix $E$ and a set of matched points among the left and right cameras. The matched points are found using SIFT and RANSAC but the motion model is described by the E matrix instead of an homography. The E matrix maps normalized points in one image to normalized epipolar lines in the other image. For the outlier rejection, the Sampson distance [14] is used to assess the inliers.

After RANSAC, the unknown extrinsic parameter are estimated from the set of all inliers using a parameterization for the E matrix based on the rotation angles $\alpha, \beta, \gamma$ and two angles that define the direction of the baseline in spherical coordinates $\theta, \phi$. The baseline $T$ can be obtained as $T = [\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta]^T \cdot r$, and $r$ is the baseline distance. The reason for using spherical coordinates is that the length of the vector $T$ cannot be determined just from matches between images [14]. However, this baseline distance is assumed to be known, and can in fact be measured relatively precisely in the field.

The extrinsic parameters are estimated by

$$(\alpha, \beta, \gamma, \theta, \phi) = \arg \min \sum_{i} d^2(x_{t}^{i}, E \cdot x_{t}^{i}) + d^2(x_{t}^{i}, E^T \cdot x_{t}^{i})$$

where $E = [R_{3D}(\alpha, \beta, \gamma)]^T \cdot [t(\theta, \phi)]_x$ and $t = [\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta]$, $d(\cdot, \cdot)$ is the perpendicular distance between a point and an epipolar line.

IV. RESULT

The experiment was done by using Matlab and two unsynchronized Gopro cameras placed in underwater housings attached to the opposite ends of a straight bar. The length of the bar is known and both cameras are facing front. The camera orientation can be measured by only using the resulting images from the video recorded by each camera without a calibration grid. The sequences were taken in the test pool using two baselines: a narrow baseline of only 35 mm and a wider one of 350 mm employing single camera housings. This sequence was used for the synchronization test and for both the standard calibration and for the extrinsic calibration from point matches. Another sequence was acquired at sea, with a similar setup, but with a shorter baseline of 95.5 mm. These sequences are summarized in Table I.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Baseline (mm)</th>
<th>Housing</th>
<th>Synchronized</th>
<th>External Calibration</th>
</tr>
</thead>
<tbody>
<tr>
<td>PoolNarrow</td>
<td>35</td>
<td>Modified GoPro Stereo</td>
<td>Yes</td>
<td>Standard toolbox</td>
</tr>
<tr>
<td>PoolWide1</td>
<td>350</td>
<td>Separated</td>
<td>No</td>
<td>Standard toolbox</td>
</tr>
<tr>
<td>PoolWide2</td>
<td>350</td>
<td>Separated</td>
<td>No</td>
<td>From matches</td>
</tr>
<tr>
<td>Sea</td>
<td>95.5</td>
<td>Separated</td>
<td>No</td>
<td>From matches</td>
</tr>
</tbody>
</table>

TABLE I

SUMMARY OF COLLECTED STEREO SEQUENCES
A. Synchronization with a flashing light

The frame differences between unsynchronized cameras have to be known in order to use the stereo rig. In the synchronization by using a flashing light, a peak happens when a strong light is detected within the frame as seen in figure 1. The peak happens in different frames because the right camera started recording before the left camera. The frame difference can be estimated from the peak location.

![Fig. 1. Peak in left (a) and right (b) camera image sequence.](image)

B. Synchronization from motion consistency

Another method to do synchronization of two cameras is using motion. Figure 2 depicts the angular velocities of $\alpha, \beta, \gamma$ and $\theta$ that were estimated from homographies computed from sequence PoolWide1. The signal describes the time differences or lags between sequences.

![Fig. 2. Unfiltered angular velocity](image)

Figure 3 shows the result of the normalized cross-correlation of the signals from Figure 2. The correlation peaks are clearly visible and consistent among themselves. Although not used in this research, a criterion to evaluate the quality of the lag estimate for various angles would be to analyze the width of the correlation peak. Wider peaks would suggest more uncertain correlations. From the plots, the $\theta$ angular velocity is the most distinctive regarding the ratio of the largest peak to the secondary peaks. This suggests that a saccadic motion in any direction is suitable to be detected by the $\theta$ angular velocity. The peak within the signal in Figure 3 is later used to find the frame difference.

![Fig. 3. Result of the normalized cross-correlation of the smoothed velocity signals](image)

In Figure 2, it can be seen that the signal is noisy. This noise is smoothed by Savitzky-Golay filter. When the SIFT algorithm finds matches between current and next frames, it also detects incorrect points which are usually found in water surface reflections. These problems can be reduced by lowering the threshold of acceptance of inliers, but could result in decreasing the number of correct matches. However, these incorrect matches are unlikely to happen in practice. Because when the cameras are used to record video of coral reefs and fishes, they normally face downward where water surface reflections are unlikely to happen.

C. Stereo extrinsic calibration from image registration

The extrinsic calibration was tested in stereo pairs of PoolWide2 right before the camera tilt, after the camera tilt and on a sea sequence. The set inliers are found by matching using the essential matrix. The results of the extrinsic calibration are shown in Figure 4, 5 and 6. As a criterion to evaluate and visualize the quality of the calibration, the stereo rectification was computed. A necessary condition for the calibration to be considered successful is that the rectified stereo pairs must have corresponding horizontal scan lines.

Figure 4 shows a good result of the standard stereo calibration where misalignments of the scan lines are barely visible. The center figure illustrates the optimization initialization values which were set to be far apart from the correct ones. The optimization is able to find a set of calibration values where the scan lines match perfectly.

Figure 5 presents the same arrangement for the part of sequence where one camera was rotated. As expected, the top rectification is clearly wrong. The starting point for the optimization was, in this case, set to represent a perfect fronto-parallel configuration where all five angles of the essential matrix parameterization are set to zero. The result of the estimation process is also quite good, with no noticeable misalignments. Finally the result for the sea sequence is presented in Figure 6, also attaining good results.

A final set of results are reported for illustrating the effects of the baseline length in measuring distances between 3D points in the scene. Table II corresponds to the GoPro stereo
housing with the baseline of 35 mm. Stereo pairs were taken approximately at 1 meter intervals from roughly 1 to 5 meter distance. Several distance measurements were done to simulate objects of different sizes that are visible in a plane approximately perpendicular to the optical axis (fronto-parallel-plane).

Table II is the accuracy test for the PoolNarrow sequence with the object viewed from near to far. 616 mm is the width or height of 11 squares in the chessboard. 224 mm is the width or height of 4 squares and 56 mm is the width or height of 1 square. Table III shows the accuracy test for the PoolWide1 with the object viewed from near to far. 480 mm is the real height of a clay pot. The standard deviations marked with an (*) have no statistical meaning, since for these cases the distribution of the uncertainty of the distance measurements is clearly non-gaussian.

To evaluate the uncertainty and therefore the validity of the distance measurements, a Monte Carlo test was implemented to compute the distribution of the distance measurements under the effect of noise in the location of the image coordinates of the points marked by a human operator. For this test a conservative value for the uncertainty of 4 pixels standard deviation was assumed, which corresponds to 1/500 of the image width. Using the two pairs of correspondences clicked by the user, two 3D points are found by triangulation. The length in 3D of this segment is reported as the distance. The two 3D points are reprojected into the images. Zero mean gaussian noise is added to these point projections and a new 3D distance is computed. This process is repeated 1000 times, standard variation of the distance is computed along with the histogram.

It can be easily seen that the errors and the uncertainties do not scale well with the increasing distance from the camera to the scene. For distances of more than 4 meters the errors grow significantly and the resulting measurements are meaningless. The distribution of the uncertainty becomes clearly non-gaussian even at short distances and is better approximated by a gamma distribution. The same behavior can be observed for the PoolWide1 sequence, although considerably less severe, due to the wider baseline which is ten times larger than in the previous case.

V. CONCLUSIONS AND FUTURE WORKS

A. Conclusions

Coral reef studies, fish sizing and their population distribution assessment are some of the documentation activities that can benefit from a simple and inexpensive stereo imaging system. This research has presented and tested techniques to achieve the goal to perform the calibration and synchronization of two inexpensive cameras operating independently.

Camera synchronization and extrinsic calibration was performed using computer vision techniques with very promising preliminary results. The camera synchronization was solved in this work in two different ways but they are limited to the fact that only two frames are computed at a time thus it cannot be more precise than 1/30 second. This is not a problem when the camera is static, or when the object being measured is moving slowly. Inaccuracy may be induced in the measurements when the speed of the image motion is high, for example when the camera is panning quickly. While the main objectives of the work were attained, by demonstrating the use of a stereo system that does not require an extrinsic calibration procedure in the field, the performance of the method is dependent on the number of inliers that are found for computing the essential matrix. In this work it was illustrated with just results obtained from a single stereo pair. However, the same approach could be used with results from multiple stereo pairs as long as the geometry of the stereo system is constant during the acquisition of those stereo pairs.

![Original Extrinsic Calibration](image1)

![Starting Point for Extrinsic Estimation](image2)

![Final Extrinsic Estimation](image3)

Fig. 4. Rectified image of PoolWide2 sequence before tilting of the right camera. The top figure show the result of the standard stereo calibration. The center illustrates the starting values for the extrinsic estimation from matches. The values were deliberately set to be wrong. The lower figure corresponds to the final values of the estimation.

<table>
<thead>
<tr>
<th>No.</th>
<th>Real distance (mm)</th>
<th>Measured distance (mm)</th>
<th>Standard Deviation of distance (mm)</th>
<th>Distance error (mm)</th>
<th>Estimated Distance of object to camera (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>616</td>
<td>587.4</td>
<td>27</td>
<td>-28.6</td>
<td>1.4</td>
</tr>
<tr>
<td>2</td>
<td>224</td>
<td>232.2</td>
<td>17.3</td>
<td>8.2</td>
<td>1.4</td>
</tr>
<tr>
<td>3</td>
<td>56</td>
<td>53.3</td>
<td>19.6</td>
<td>-2.7</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>616</td>
<td>570.4</td>
<td>61.3</td>
<td>-45.6</td>
<td>2.6</td>
</tr>
<tr>
<td>5</td>
<td>224</td>
<td>223.1</td>
<td>84.7</td>
<td>0.9</td>
<td>2.6</td>
</tr>
<tr>
<td>6</td>
<td>56</td>
<td>48.6</td>
<td>104.5</td>
<td>-7.4</td>
<td>2.6</td>
</tr>
<tr>
<td>7</td>
<td>616</td>
<td>561.7</td>
<td>171.7</td>
<td>-54.3</td>
<td>3.9</td>
</tr>
<tr>
<td>8</td>
<td>224</td>
<td>248.3</td>
<td>340.6</td>
<td>24.3</td>
<td>4.7</td>
</tr>
<tr>
<td>9</td>
<td>56</td>
<td>641</td>
<td>392.1</td>
<td>5</td>
<td>4.8</td>
</tr>
<tr>
<td>10</td>
<td>616</td>
<td>863.8</td>
<td>550.9 (*)</td>
<td>247.8</td>
<td>5.3</td>
</tr>
<tr>
<td>11</td>
<td>224</td>
<td>486.1</td>
<td>627.8 (*)</td>
<td>262.1</td>
<td>5.9</td>
</tr>
<tr>
<td>12</td>
<td>56</td>
<td>212.7</td>
<td>661.2 (*)</td>
<td>156.7</td>
<td>6.1</td>
</tr>
<tr>
<td>13</td>
<td>616</td>
<td>2581.6</td>
<td>1796.5 (*)</td>
<td>1914.6</td>
<td>7.6</td>
</tr>
<tr>
<td>14</td>
<td>224</td>
<td>850.9</td>
<td>663.3 (*)</td>
<td>279.9</td>
<td>7.6</td>
</tr>
<tr>
<td>15</td>
<td>56</td>
<td>1911</td>
<td>1598.9 (*)</td>
<td>1855</td>
<td>7.8</td>
</tr>
</tbody>
</table>

TABLE II

ACURACY TEST FOR POOLNARROW
B. Future Works

As future work, further testing is required for validating the approach in terms of error accuracy also to detect when the result are inconsistent based on the analysis of the reprojection errors. When estimating the extrinsic calibration, the modeling of the rolling shutter effect and the estimation of the parameters for the end model of the rolling shutter effect and the correction of this effect.

VI. ACKNOWLEDGMENTS

This work was conducted in University of Girona, Spain.

REFERENCES

Classification and Recognition of Traffic Signs

Farhat, Dr. Yohan Fougerolle

Abstract—Traffic sign is the main component in the highway that aims to organize and provide information to the user, for road safety. The given information can be as a guidance, prohibition, warning, etc. A development of an application that has an ability to automatically detect and translate the meaning of the sign can help increase the awareness of traffic sign and road safety. One field of technology in this regard is the Traffic Sign Recognition (TSR). In this paper, implementation of TSR especially for the classification stages is performed. Combination of SIFT and HOG to extract the features, Singular Value Decomposition (SVD) method to reduce and keep only the significant features, Bag of Words (BoW) to represent the extracted features to the Support Vector Machine (SVM) as a classifier. The accuracy obtained was 84%. This is because the lighting factor. Almost all of the misclassified signs are the signs with a dark view or get reflection of the light in large numbers.

I. INTRODUCTION

Traffic signs are one of the most important components in traffic environments because it helps to create a controlled and safe traffic. The main focus is to improve the ability to identify the traffic sign which in turn will improve road safety. A TSR system is composed of two principal steps: the detection and the classification. The detection stage aims as automatically detecting the traffic sign by doing segmentation and various pre-processing operations. The results of this stage will be forwarded to the classification stage to determine the type and the information contained in the sign. In the implementation, there are some major problems:

1) The signs are captured in various
   • Weather conditions, such as fog or rain.
   • Lighting / illumination conditions: day and night.

2) Geometrical issues
   • Occlusion.
   • Rotation.
   • Different view point.
   • Scale distortion.

3) Self Similarity
   Can be happen for signs from different categories (intra-class similarity) and for the signs from the same category (inter-class similarity).

4) Similarity between some surrounding objects and the signs (in terms of color and shape).

II. RELATED WORK

Considering the importance of Traffic Sign Recognition, there are many researches have been done using various methods.

In [1], the authors perform classification by using the Bag of Words (BoW) approach, where the Scale Invariant Feature Transform (SIFT) features that have been extracted from the training images are stored in the ‘Individual Dictionaries’. Then they combine all the individual dictionaries and become ‘Modular Codebook’. This is useful if there is a new class of the data, then recomputed of the modular codebook is not needed. The testing is conducted by comparing the features with the modular codebook and at the same time the histogram is generated. This histogram is used as input to a linear SVM (One vs. All SVM). Scale Invariant Feature Transform (SIFT) is selected because it is scale and rotation invariant. Recognition results is 97%.

Classification stages in [2] similar to [1], the only difference is that they use SVM with RBF kernel (using LIBSVM library). While in [3], linear CSVM is used. [4] utilizing the SVM with Gaussian kernel. Test is only performed on the image with the same color and shape. SVM with Gaussian kernel is also used in [5], but the classified sign is simply the circular traffic sign.

In the [6], classification of the detected traffic signs is performed by using information of the moment. According to Fleyeh, moments are useful for feature extraction because provide information about the geometrical features and represent the global characteristic of the shape of an object in the image. Fleyeh performs various comparisons between the features, different kernels type and obtain that the best features for traffic signs recognition was Legendre moments. Similar in [12], but Haar Invariant Features yields the result of 97.77%.

Alberto Broggi in [7], conducted sign classification by using NN based on shape and color.

Normalized Cross Correlation is used in [10] and with a variety of different color such as in [9] HSV color space was used as it separated the color and intensity, [13] and [8] using a gray scale image because of Hue component in the HSV color channel is affected by weather, age and the distance between the sign detection tool.

[15] perform template matching in the form of traffic sign that has removed the background. SURF descriptors of the detected traffic sign’s pictogram in [16] is used in classification stage. According to a study of Juan et al. [11], SURF is more robust to illumination changes. The recognition rate was 97.72% Achieved.
III. MATHEMATICAL BACKGROUND

A. Image Gradient

Image gradient expresses the directional change in an image (the change of intensity / color). Gradient of each image point (image pixels) is represented in 2D vector, that represents:

1) The magnitude shows the intensity changes along the vertical (Y) or horizontal (X) direction (expressed as the length of the vector).

\[
\begin{align*}
(dy) &= A(x, y - 1) - A(x, y + 1) \\
(dx) &= A(x + 1, y) - A(x - 1, y)
\end{align*}
\]

\[
\|A(x,y)\| = \sqrt{(dx)^2 + (dy)^2}
\]

- A(x,y) is an pixel in image A
- x is the position of pixel A in X coordinate
- y is the position of pixel A in Y coordinate
- dx is the measurement of the change of pixel value in X coordinate
- dy is the measurement of the change of pixel value in Y coordinate

2) Orientation / direction of the gradient shows the direction of the changing in image (in angle). The values of magnitude that have been obtained previously are used to calculate the orientation value (\(\theta\)).

\[
\theta = \tan^{-1}\left(\frac{dy}{dx}\right)
\]

B. Features Descriptor

Features are properties of an image.

1) Histogram of Oriented Gradients (HOG): HOG is a local feature descriptor developed by Dalal and Triggs [6]. HOG will divide the image into small windows / cells (small region, for further feature extraction is done in this area). In every region / cell, the gradient calculation will be performed to all the pixels and represented the obtained \(\theta\) values into a histogram (with certain number of bins). At the end, all the obtained histograms from all cells will be merged and will be a final feature extracted from the image. The detail workflow of HOG computation can be seen in Figure 1(a).

2) Scale Invariant Feature Transform (SIFT) Descriptor: Has been developed by Lowe [14] and transforms an image into scale-invariant coordinates. The obtained features invariant to various geometric condition (scale, rotation), noise, affine distortion and partially invariant to change in lighting (illumination) and different viewpoint. In contrast to the HOG that extracting the features on all parts of the image, SIFT extracts features only on some parts of the images that are considered to have important information. SIFT computation stages can be seen in Figure 1(b).

Determining the location and scales can be performed by using the convolution of Gaussian functions with the input image.

\[
G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}
\]

\[
L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)
\]

- \(L(x,y,\sigma)\) = Scale Space of an image
- * is convolution operation

Keypoint location detected by using the difference of Gaussian function (two nearby scales) convolved with the image in scale space. This is performed incrementally.

\[
D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)
\]

\[
D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)
\]

- \(D(x,y,\sigma)\) is the Difference of Gaussian function(DoG).
- k is a constant multiplicative factor.
- \(\sigma\) is the sigma value of gaussian function.

Keypoint is obtained by comparing a pixel with 27 neighbors pixels (8 pieces pixel on the same scale, 9 pieces of pixels on the scale above and 9 pixels on the scale below). In the end the local maxima and minima of \(D\) (x, y, \(\sigma\)) is obtained as keypoint.

C. K-Means Clustering

Is a method to do partition of N observations into K clusters in which each observation belongs to the cluster with the nearest mean (Figure 2).

\[
Cluster = \sum_{i=1}^{k} \sum_{x_j \in S_i} ||x_j - \bar{X}||^2
\]
K (Clusters),

Data set

Randomly choose K data as centroids / seeds

For loop 1 = 1: K

Grouping the data based on the minimum distance to the seeds (Euclidean Distance)

End of loop 1

Assign new seeds based on the min distance with prev seeds

End

Y

N

Y

N

Fig. 2. K-means workflow

D. Bag of Words (BoW)

Bag of Words is a commonly used method in the classification. In this method, an image can be treated as a document and the extracted features will be considered as words. K-Means clustering is performed on the set of features and yields codebook. Frequency / occurrence calculation between features with the codebook is performed. The calculation result will be obtained in the form of frequency histograms. This histogram is used as a feature for training a classifier.

E. Support Vector Machine (SVM)

Classification process begins by generating and finding the best and optimal model / hyperplane that can separate the extracted features linearly into the appropriate classes. More precisely, the hyperplane can be used as a decision function (models) that obtained from the training process, the one which is intended as a boundary between classes. Optimal hyperplane is defined as a candidate hyperplane which has maximum distance with the data located at outer boundary from each class. This distance is called Margin, while the data located at the outer boundary from each class is called the Support Vectors. If in the input space, the data cannot be separated linearly, then mapping the inputs into the higher dimensional space (Feature Space) is needed with the help of kernel techniques. This mapping approach is known as "Kernel Trick". This paper uses RBF as the kernel.

\[
RBF : K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \gamma > 0
\]  

The chosen kernel for SVM is RBF it has been widely proven as a good kernel and can handle if the relation between class labels and attributes is non-linear (by nonlinearly maps samples into a higher dimensional space). In addition, if compared with the polynomial kernel whose value can be zero or infinity (influenced by the degree), RBF does not has that such numerical difficulties. In doing the mapping, RBF kernel requires gamma parameter \((-\gamma)\). The gamma \((-\gamma)\) parameter is obtained by doing cross validation. The best parameters obtained from cross validation are used to train the whole of system and get the model.

IV. IMPLEMENTATION

Figure 3 shows the general flow of the developed system. This classification system uses a combination of SIFT and HOG features. Both of these features are extracted from each image, and the significant features are selected using SVD. Then, the frequency quantization is performed, based on each codebook.

The obtained histogram for SIFT and HOG then combined and is used by SVM to train and generate the model. This model will be used for a whole system to classify traffic sign image (Figure 4).

V. RESULT

The training and testing stages have been conducted on a total of respectively 2480 and 1215 images from 31 classes of traffic signs with a various conditions (ranging from lighting, view point and size). Figure 5, 6 and 7.
To facilitate the visualization and the delivery of information of the classification result, Confusion Matrix is used. Confusion Matrix is a visualization of the performance of the classifier to classify correctly the traffic sign images. Each column represents the predicted class and the row is the actual class. From the matrix, it can be seen the number of images from one class that misclassified or classified correctly. For the misclassified images, distribution of the prediction class can be determined easily.

**TABLE I**

A Confusion Matrix of the developed classification stage

<table>
<thead>
<tr>
<th>Predicted Class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Class</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>26</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>29</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

From the testing stage that has been conducted and the result presentation in the confusion table, it can be seen that there are many misclassified traffic signs. From the analysis on the obtained result, can be grouped:
1) Misclassified
Signs that are belongs to this group are the signs which have a extreme dark view and which get the reflection of light in a hefty amount. Can be seen in Figure 8.

![Fig. 8. Images from class 15 that are misclassified due to the lighting effect.](image)

2) Correctly classified
Signs that are belongs to this group are the signs with various conditions: the view is not really dark, blurred, signs with different view point, and has a little bit reflection. Figure 9 and Figure 10.

VI. CONCLUSION
From the conducted testing, the accuracy level of the developed classification system to correctly classified the signs is 84 %. This is because, 16 % are mostly images with extreme lighting conditions (dark view and get high reflection by the light). As for other conditions, such as a different view point, a different scale and non extreme lighting conditions, the system proved able to correctly classify.

REFERENCES
Fig. 9. Images from class 15 with various conditions that are correctly classified.

Fig. 10. Images from class 9 with various conditions that are correctly classified.
Implementation of a Behavioral-based Navigation System for Mobile Robot BIGBOT

Isram Rasal, Xavier Cufi and Albert Figueras

Abstract—There are many different types of disasters such as earthquakes, flood, building collapses, landslides or volcano eruption. During emergency situations, and especially in urban disaster, usually policeman, fireman and medical assistance are deployed. They need to cooperate with each other to save human lives, protect the structural infrastructure, and evacuate victims into a safe place. An Urban Search and Rescue (USAR) operation involves the location of these victims, their medical stabilization and their extraction into the safety zone for further treatment.

We designed and implemented a novel approach to improve the USAR performance. The novelty of this paper is to develop Behavioral-based Navigation System for mobile robot, which by using mobile robot's visual cognitive, the mobile robot can interact with the USAR trained dog.

3D object colored tracker is implemented and tested in practice using C++, Python and Robot Operating System (ROS). 3D object colored tracker has been developed as an independent ROS module using CamShift by OpenCV library and Voxell Grid filter by Point Cloud Libraries, it receives image messages from a Kinect image stream, process them and produce control commands. The control commands are then published as ROS messages. A velocity controller module will receive these messages and execute them. The functionality of 3D object colored tracker and teleoperation mode are tested in a Agent Research Laboratory room and our implementation shows satisfactory results.

I. INTRODUCTION

A. Background

There are many different types of disasters such as earthquakes, flood, tsunami, hurricane, and they lead to different disasters such as building collapses, landslides or crater. During emergency situations, and especially in urban disaster, usually policeman, fireman and medical assistance are deployed. They need to cooperate with each other to save human lives, protect the structural infrastructure, and evacuate victims into a safe place.

An Urban Search and Rescue (USAR) operation involves the location of the victims, their medical stabilization and their extraction into the safety zone for further treatment. Urban Search and Rescue. USAR teams normally uses trained dogs as a friend to find hidden victims (figure 1). Trained dogs are very helpful because they have high mobility, speed and detection ability. Whereas, they need direction and supervise by the USAR teams. In some situations, for instance, in a presence of dangerous gas, USAR teams and trained dogs are in danger situation, they should leave away the area. All these characteristics mean that autonomous or semi-autonomous robots can be a very useful tool in USAR operations, giving appropriate support to all the components of USAR teams. Robots have a high sensorial capability, can collect and interpret data in a precise way, also can operate in hazardous environments.

Fig. 1. Sweden MSB SWIFT USAR canine team [12]

B. Problem Definition and Project Goals

Anna Bosch et all in [3], proposed a new USAR scenario that uses mobile robots to improve the efficiency of USAR teams in hazardous environments. This research [3] describes the MATE project [8]. MATE project consists of several institutions in European area, with the participation of University of Girona and University of Burgundy. The main challenge of MATE project is the dog-robot-human interaction that: (i) to give visual cognitive and reasoning abilities to the robot in order to let the robot autonomously or semi-autonomously interact and cooperate with the dog according to its behavior and environmental conditions, and (ii) to specify train a dog to correctly accept and interact the robot.

Some researchers from Agents Research Laboratory (AR-Lab) in University of Girona has developed mobile robot BIGBOT that can operate in difficult terrains such as sands, gravels, and rocks. The first version of BIGBOT is manually controlled by operator (human) through Labview interface, there are no ability to run in autonomous navigation by using visual information.

In brief, the goals of the project which described in this paper are:

- To develop autonomous navigation system on mobile robot BIGBOT which based on visual cognitive to detect
and follow a specific trained dog and can deal with occlusion.

- To develop semi-autonomous navigation system on mobile robot BIGBOT by using teleoperation mode.
- To implement autonomous and semi-autonomous navigation of the BIGBOT robot into a new software architecture.

The rest of this paper is organized as follows: Section II presents current research relating to USAR robots, Section III describes the development of Behavioral-based Navigation System, Section IV describes software design, Section V describes the experiments and results, and the last is Section VI, describes the conclusions and future works.

II. RELATED WORKS

North Carolina State University and several team of researchers from around the United States are working on Smart Emergency Response System (SERS) project. They trying to develop dog-robot-human interaction, using a similar concept as MATE's project. By using a combination of ground and aerial autonomous vehicles like drones, humanoids, human-operated telerobots, and trained SAR dogs equipped with Canine Assisted Robot (CAR) to save human lives as much as possible in an emergency situation [1](figure 2).

![Fig. 2. Several robots and trained dog by North Carolina State University, United States of America [1]](image)

The Network-Centric Applied Research Team (N-CART) in Department of Computer Science at Ryerson University, which involve in MATE project, has been defining the concept of Canine Augmented Technology (CAT). CAT consists of a sensor, actuation and communication package which dressed on canine USAR dogs [9] (figure 3).

![Fig. 3. Trained dog equipped with Canine Augmentation Technology (CAT) [11]](image)

III. DEVELOPMENT OF BEHAVIORAL-BASED NAVIGATION SYSTEM FOR MOBILE ROBOT BIGBOT

A. Behavior-based Robotics

Behavior-based robotics is a methodology or approach for designing autonomous agents and robots. The behavior-based methodology aim is to develop methods for controlling artificial systems, usually physical robots, but also simulated robots and other autonomous software agents. There are some basic principles that have been used by researchers in behavior-based robotics. Parallelism, modularity, situatedness/embeddedness, and emergence are the key success of this methodology [13].

B. Behavioral-based Navigation System for Mobile Robot BIGBOT

In order to achieve our project goals, we designed Behavioral-based Navigation System for mobile robot BIGBOT, which consist of three behaviors:

- **Looking for colored patch behaviour**: This behaviour is used to find an object by its color. If we look at regular USAR trained dog, usually they wear a special cloth with a conspicuous color (usually red or orange).

- **Following colored patch behaviour**: This behaviour can run if looking for colored patch behaviour is running well. This behaviour is to follow desired colored object and keeping a safe distance. The inputs of this behavior are RGB camera and IR camera. The algorithm computes all distances and creates a 3D space. The direction of the robot must be taken to keep distance with the object which detected by the IR camera.

- **Teleoperating behaviour**: This behaviour means that mobile robot is manually controlled by human. This behavior becomes active in three specific situations. The first is when looking for colored patch behaviour is fail. The second is when looking for colored patch behaviour is not activated. The last is when performing USAR operation with trained dog to find a victim by using following colored patch behaviour, and the moment the trained dog barking or giving a signal that victim is found and then the operator (human) in control room take the control of the robots platform.

IV. SOFTWARE DESIGN

We proposed 3D colored object tracker which based on Continuously Adaptive Mean Shift Tracking and Voxel Grid filter in ROS platform.

A. Robot Operating System (ROS) as Software Platform

Robot Operating System (ROS) is a framework which is commonly used in robotics area. ROS was designed to meet a specific set of challenges encountered when developing a large-scale service robots as part of the STAIR project at Stanford University and the Personal Robots Program at Willow Garage, but the resulting architecture is far more general than the service-robot and mobile-manipulation domains [14]. ROS is distributed under Berkeley Software
Dense regions (or cluster) of maximum pixel density or maximum number of points [4]. It is illustrated in the figure 4.

**B. Continuously Adaptive Mean Shift Tracking**

The Mean Shift algorithm is a robust, non-parametric technique that climbs the gradient of a probability distribution to find the mode (peak) of the distribution. Mean Shift was first applied to the problem of mode seeking by Cheng in [6]. Mean Shift considers feature space as an empirical probability density function. If the input is a set of points then Mean Shift considers them as sampled from the underlying probability density function (pdf). If dense regions (or cluster) are present in the feature space, then they correspond to the local maxima of the pdf [7].

The intuition behind the Mean Shift is simple. Consider we have a set of points. It can be a pixel distribution like histogram backprojection. We are given a small window or may be a circle, and we have to move that window to the area of maximum pixel density or maximum number of points [4]. It is illustrated in the figure 4.

![Mean Shift basic approach](image)

In figure 4, the initial window is shown in blue circle with namely "C1". Its original center is marked in blue rectangle, namely "C1o". But if we find the centroid of the points inside that window, we will get the point "C1r" (marked in small blue circle) which is the real centroid of window. Surely they do not match. So move our window such that circle of the new window matches with previous centroid. Again, find the new centroid. Most probably, it will not match. So move it again, and continue the iterations such that center of window and its centroid falls on the same location or with a small desired error. So finally what we obtain is a window with maximum pixel distribution. It is marked with green circle, namely "C2". As we can see in figure 4, it has maximum number of points.

Continuously Adaptive Mean Shift (CamShift) is primarily intended to perform efficient head and face tracking in a perceptual user interface by Gary Bradsky [5]. It is based on an adaptation of Mean Shift algorithm that given a probability density image, finds the mean (mode) of the distribution by iterating in the direction of maximum increase in probability density. The main difference between the CamShift and the Mean Shift algorithm is that CamShift uses continuously adaptive probability distributions. Mean Shift is based on static distributions, which are not updated unless the target experiences significant changes in shape, size or color. Since CamShift does not maintain static distributions, spatial moments are used to iterate towards to mode of the distribution. This is in contrast to the conventional implementation of the Mean Shift algorithm where target and candidate distributions are used to iterate towards the maximum increase in density using the ratio of the current (candidate) distribution over the target.

The CamShift algorithm can be summarized in the following steps [5]:

1. Set the region of interest (ROI) of the probability distribution image to the entire image.
2. Select an initial location of the Mean Shift search window. The selected location is the target distribution to be tracked.
3. Calculate a color probability distribution of the region centred at the Mean Shift search window.
4. Iterate Mean Shift algorithm to find the centroid of the probability image. Store the zeroth moment(distribution area) and centroid location.
5. For the following frame, center the search window at the mean location found in Step 4 and set the window size to a function of the zeroth moment. Go to Step 3.

OpenCV (Open Source Computer Vision) provide CamShift library, we can utilize it to bring our **looking for colored patch** behaviour into reality.

**C. Voxell Grid Filter**

Point Cloud Library (PCL) has a downsampling function, its called voxel grid filter. A voxel grid filter serves as an approximation of the real surface and is used to separate the shape functions into more descriptive histograms representing point distances, angles, and areas, either on the surface, off the surface, or both [2].

We proposed a new system based on ROS framework to be implemented in mobile robot BIGBOT in order to give a robust solution. In our design, ROS as platform in higher level that handle our designed behavior. By using Kinect and Joystick as input, ROS provide two main task: autonomous navigation and semi-autonomous navigation. Then, ROS will send the velocity (angular and linear) data to BIGBOT on lower level (controller board). Our design (see figure 5) implies to the new configuration. We call it the second generation of mobile robot BIGBOT.

**V. Implementations and Results**

In the experiment, we use a low cost dummy trained dog which made by a red box that mounted on the SMALLBOT (figure 6). The experiment was performed in the Agent Research laboratory room which has a flat floor surface.
We implement a Continuously Adaptive Mean Shift (CamShift) tracking using OpenCV library. The way of our code can run are described in three major steps:

1. From openNI stack we can obtain RGB color image frame from Kinect sensor. The input RGB color image (figure 8(a)) is converted into HSV color space. This step is done by using \textit{cv2CvtColor} function (figure 8(b)).

2. The HSV image data is used in conjunction with a color histogram in a process called Back-Projection, which essentially produces a color probability image from the input RGB color image. The Back-Projection image encodes for each pixel, its probability of belonging to the color probability distribution represented by the histogram. The histogram itself initialized by sampling a representative area of one frame, we specified manually by the progress bar (figure 8(c)). OpenCV provides data structures for histograms and associated functions, this step done by using \textit{cv2CalcBackProject} (figure 8(d)).

3. Finally, the core of the CamShift algorithm itself is applied, coded as a call to the \textit{cv2CamShift} function. Given an initial ROI window (figure 8(e)) and a Back-Projection image, the function returns the bounding box of the most probable detection area in the image, and a size and orientation estimate of the distribution (figure 8(f)). The bounding box is used to infer the initial search window in the next input back-projection image. The initial ROI window data (input) is related to the detected area (output) in the previous frame. In an asynchronous approach, this information can be encoded as the last known detection result (bounding box), which is persistent information updated after the processing of each new frame, thus forming a feedback loop, as known as tracking (figure 8(g)).

\subsection*{B. Implementation of Tracking for 3D Colored Object}

In above, we have implemented how to use OpenCV to track colored object by using CamShift algorithm. The result is a bounding box / region of interest (ROI) that follows the desired colored object and the ROI published on the ROS topic /roi. Our Kinect is mounted on a mobile robot BIGBOT, we use the x-offset coordinate of the /roi to keep the desired colored object centered in the field of view by rotating the robot to compensate for the offset. In this way, the robot will track the desired colored object as it moves to the left or right in front of the camera. On the other hand, we use z-axis of point clouds which published by Voxell Grid filter. In this way, the robot will track and keep the safe distance to the desired 3D object as it moves forward or backward in front of the Kinect.

We sample all the points in the depth clouds that lie within a desired search box in front of the mobile robot BIGBOT. From those points, we compute the centroid of the z-axis region. If there is an object in front of the mobile robot BIGBOT, the z-coordinate of the centroid tells us how far away they are. From these z-axis point numbers we can compute an appropriate linear velocity to keep the robot near the object. At the same time, we compute the x-axis displacement of the target from the center of the
Fig. 8. Implementation of Continuously Adaptive Mean Shift Tracking

Fig. 9. Result of 3D colored object tracking

Fig. 10. CamShift tracking with occlusion
camera image. Recall that the $x_{\text{offset}}$ field in an ROI message specifies the x-coordinate of the upper left corner of the region, so to find the center of the region we add half the width. Then to find the displacement from the center of the image we subtract half the width of the image. With the displacement computed in pixels, we then get the displacement as a fraction of the image width. Finally, we compute the angular speed of the robot to be proportional to the displacement of the target. If the target offset does not exceed the $x_{\text{threshold}}$ parameter, we set the angular velocity to zero. From these two velocity parameters (angular and linear) we can compute an appropriate Twist cmd.vel message to keep the robot following the object.

When controlling a mobile robot, it is always a nice idea to set a maximum speed. Setting a minimum speed as well can make sure that the robot does not struggle against its own weight and friction when trying to move too slowly. We do not want the robot to be draining its battery, just because chasing every small movement of the target. So we set a lower threshold on the displacement (in x, y and z axis) of the target to which the robot will respond.

C. Results

Our implementation shows satisfactory results, mobile robot BIGBOT can follow SMALLBOT by using 3D colored object tracker (see figure 9). The BIGBOT try to chasing the SMALLBOT but keep a safe distance and moves as well as our design, if the SMALLBOT go to forward right, the BIGBOT also moves to forward right, but the movement its depend on our safe distance and the gain parameter. Remember that turn right and left movement is depend on the ROI boundary and forward and backward movement is depend on the object position in front of Kinect.

We also test CamShift tracker with occlusion. While CamShift tracks desired color patch, and some object obstruct the desired object, at one moment CamShift track the wrong object but then CamShift recover it and track our desired color patch as well (see figure 10).

VI. CONCLUSIONS AND FUTURE WORKS

A. Conclusions

There are exists two unusual projects in USAR robotics. SERS and MATE project bring a new approach on USAR scenario, which combining robots, dogs and humans into USAR operation. However, MATE project has some advantages, one of the advantages is to linking trained dogs and robots by using the robot’s visual information. It is a novel approach on dog-robot behavior which presented in MATE proposal. The deep research on trained dog and USAR robotics has lead to several new technologies such as Canine Augmented Technology (CAT) and Canine Remote Deploymen System (CRDS).

Belong to the theory of Behavior-based Robotic, we defined three behaviors called Behavioral-based Navigation System for mobile robot BIGBOT. These kind of behaviors are the main approach to realize our project goals. Real time experiments with a low cost dummy trained dog showed that combination of Continously Adaptive Mean Shift algorithm and Voxel Grid filter is capable to create a robust 3D colored object tracker. This was noticed from the experiments, more precisely in speed controlling heave, the small variations of detected 3D colored object width to produce orders to the mobile robot BIGBOT to follow the desired object. Teleoperation mode is done by using a joystick as well as 3D colored object tracker.

B. Future Works

Future work would consider the extension of navigation system in autonomous and semi-autonomous navigation. Several works we are planning to do to make the behavioral-based navigation system for mobile robot BIGBOT more complete include:

- Localization
- Path planning
- Obstacle avoidance

The mobile robot BIGBOT has ability to determine its position and to navigate over specific rescue scenario considering a specific set of behavioral-based navigation system for mobile robot BIGBOT which has been developed in this project.

VII. REFERENCES

Our project (videos, images, etc) can be found on www.projectbigbot.tk

Fig. 11. scan this QR code to see our project

References


Abstract—Accurate autonomous vehicles localization and map building are core topics in computer vision and robotics applications, since the robot has no prior information about the environment around it. Moreover, the robot needs to have localization and mapping information online. In his research, we propose a novel approach for ego-motion estimation and 3D maps reconstruction from a high resolution, continuous sequence of stereo images in real-time. Our method depends on visual odometry algorithm with a robust sparse feature matching for motion estimation. Then, clouds registration process is performed upon the resulted transformations between frames. The assumptions we make are a known stereo camera intrinsic parameters and a continuous smooth camera trajectory. We employed a RANSAC base outlier rejection, then a Kalman filter to yield a robust frame to frame motion estimation in real-time. In our experiments we used a Bumblebee 2 stereo camera for image acquisition running on CPU core i7 with 3.0 GHz. The algorithm is able to adapt the environment changes and handles 2-3 frame per second with high accuracy and running-time.

I. INTRODUCTION

Nowadays, Robots have been involved in many fields in our life. One of the main prerequisites for any autonomous vehicle is to localize itself within the environment. Usually, the robot has several sensors (GPS, IMU, Camera, Wheel sensors and etc ...) in order to identify the robot location with respect to an initial frame. The information flow coming from sensors is gathered into a given representation to answer the question "How does the world around me looks like?". However, the robot still needs an answer for other question "Where am I now?". Localization is the answer for the second question by estimating the robot current pose with respect to the previous pose or to a static reference pose. Nowadays, many robotics and autonomous vehicle depend on laser scanners for navigation and pose estimation in unknown environment, because it can provide the 3D measurements directly in real time. The alternative solution for the tradition pose estimation method is Visual Odometry. In other words, the process of estimating the motion of a robot using a sequence of single or multi cameras embedded to a robot is called Visual Odometry. Recently, camera systems became cheaper, more compact and the needed computational power can be handled with standard PC. Moreover, the PC’s hardware capability nowadays is increasing dramatically with lower power consumption especially when GPU is used to process high resolution image at high frame rate in real-time. The proposed method uses a stereo camera for estimating the relative displacement between each two consecutive stereo camera poses. The yielded poses are used for registering the point clouds. Then, world 3D reconstruction is performed. The sequence is captured in urban environment. Outliers rejection step is used based on random sampling. The estimated ego-motion has 6 degrees of freedom (6DOF). The contributions of this research are: First, simple but efficient Visual Odometry strategy is proposed to obtain high quality ego-motion. Second, several thousand of features matches are computed in real-time scene flow. Third, based on yielded pose from ego motion estimation and thanks to library ELSA [8], a 3D reconstruction is obtained easily.

II. STATE OF ART

Computer Vision and Robotics communities have developed many techniques for 3D mapping and localization techniques and algorithms. The following part will cover briefly some techniques used by others: Simultaneous Localization and Mapping (SLAM) [1] [2] [3] is the process by which an autonomous vehicles starts from an unknown environment. Then, it tries to build its own map incrementally and at the same time uses this map to localize itself using only relative observations of the environment in order to compute its own location then be able to navigate autonomously. Recently, several algorithms have been developed based on visual odometry. These methods can be roughly divided into two groups, group one depends on features tracking over a sequence of images [4] and group two depends on feature matching between consecutive images [5]. In [4], they used IMU to estimate the rotation and some computer vision methods to estimate the translation by tracking salient features based on sequence of images taken by a camera continuously using KLT tracker, then try to calculate different vectors from the tracked features to extract the robot’s motion. In [5], a set of features are detected on each frame then match features between frames with applying a rejection criteria. A 3-dimensional scene cane be reconstructed via triangulation if a stereo camera is used. The iterated closest point (ICP) algorithm is often used to estimate the trajectory. Iterative Closest Point algorithm (ICP) tries iteratively to find the best transformation between source cloud and target cloud by finding temporary point correspondences and updating motion parameters until the system converges. By accumulating the transformations between each point cloud, the camera trajectory can be recovered as explained in [6] and [7]. In [4] IMU sensor is used to recover rotation angles, while in [9] inexpensive GPS is used with wheel encoders. In this research, we will depends mainly on novel visual
odometry algorithm [7] based on visual matching in stereo image sequence token in urban environment to build a 3D map and estimate camera motion from high-resolution stereo sequences in real-time.

III. METHODOLOGY

Our algorithm consists of five main stages. First, building camera model. Second, motion estimation. Third, building depth map. Fourth, 3D reconstruction. Finally, clouds registration. Each step will be covered in details in the following sections. The implementation uses the following open source libraries: OpenCV [12], Point Cloud [11], ELSA [8].

A. Building Camera Model

Camera model is represented in camera intrinsic and extrinsic matrices. In this research, we are working on Point Grey Bumblebee2 Stereo Camera. The input for this step is a set of calibration pattern images with different poses and the output is the camera calibration matrix.

1) Camera Calibration: Camera calibration pattern captured by the camera are used to solve camera model equations and find its matrix. OpenCV provide a method called stereoCalibrate, it takes as an input a vector of vectors of the calibration pattern points, which are the calibration pattern corners, and a vector of vectors of the projections of the calibration pattern points for each camera. By solving camera matrix, we get intrinsic camera matrix and distortion coefficients for each camera

\[
\begin{bmatrix}
    f & 0 & c_u \\
    0 & f & c_v \\
    0 & 0 & 1
\end{bmatrix}
\]

(1)

2) Stereo Rectification: Up on Bougnet’s algorithm stereo pair is rectified. In other words, all the epipolar lines will be parallel to each other and to the baseline. Since the rectification process can be implemented by projecting the original pictures onto the new image plane. Then, a new projection matrix is resulted for each camera \(P_l, P_r\), those matrices now represent the new rectified cameras matrices.

\[
P_l = \begin{bmatrix}
    f & 0 & c_u & 0 \\
    0 & f & c_v & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix}
\]

(2)

\[
P_r = \begin{bmatrix}
    f & 0 & c_u & T_r T_l f \\
    0 & f & c_v & 0 \\
    0 & 0 & 1 & 0
\end{bmatrix}
\]

(3)

B. Motion Estimation

Based on [7], this process consists of two main steps. First step is feature matching. Then, ego-motion estimation. The feature matching is responsible for extracting features between the current and previous stereo pairs and the ego-motion estimation is responsible for estimating the transformation between the current and previous stereo pairs. The input for this step is a continuous, rectified, undistorted sequence of stereo pair images. The output is the transformation (rotation and translation) of each stereo pair with respect to initial static frame coordinate.

1) Feature Matching: The input for this step is the right and left current frame and right and left previous frame. Each image of input is filtered with 5 by 5 corner and blob masks. The filter is as show in 1. The input for this step is the right and left current frame and right and left previous frame. In order to have a robust motion estimation, stable features are needed. Each image of input is filtered with 5 by 5 corner and blob masks. After that, to faster the performance and reduce the computation complexity, a non-maximum and non-minimum-suppression [10] is used to categorize the resulted features into four groups (i.e., blob max, blob min, corner max, corner min) Bases on [7], feature matching simply is computed based on finding the minimum sum of absolute differences (SAD) error metric on comparing horizontal and vertical filter responses to each other using 11 by 11 block window. The features matching algorithm assumes that we have two stereo frames, which are the left and right images from the current frame and the left and right images from the previous frame. Then, feature matching is achieved in a circle manner. Initially, the algorithm tries to find features in current left image frame. Then, within a M by M search window, it tries to find the best match in the previous left image frame. After that, since we already have the fundamental matrix from calibration step and we have the feature points (corners and blobs extracted previously) in previous left image, we can calculate the corresponding epipolar line in previous right image. Searching for corresponding point along the epipolar line with error tolerance of 1 pixel reduces the search space. The next step is similar to first step. We find the corresponding feature in current right frame whine M by M searching window. Now, we have a feature on current right image, and we need to find its correspondence in current left image frame. Also, this step is done thanks to the corresponding epipolar line in current left image. Only circle matches are accepted. In other words, if the last feature in current right frame coincides with the first feature in current first frame, we consider this features.

2) Ego-motion Estimation: From the previous step we yielded enough number of features that validate the circle condition. From those points correspondences, the 3-dimension is estimated via triangulation. The camera motion is computed by minimizing the sum of re-projection errors. After that, a refinement step is used to optimize the resulted
The re-projection is done using camera matrix 5. Assuming correspondences, then we can estimate the 3-dimensions torroid stereo camera system and we have enough point estimation performance. To have good feature points reduction, the image is divided into several non-overlapping rectangles called buckets [14]. Each bucket contains the maximum number of feature points to express this rectangle. This strategy benefits in several ways. First of all, reducing the number of features in each rectangle reduces the computational complexity as the smaller number of features reduces the computational complexity of the algorithm which is an important prerequisite for real time applications. Second, as we are working on stereo system, so each stereo pair will be a 3-dimensions cloud of points. Then, we can note that the features are distributed along the Z axis which is the roll axis. The bucketing for features distribution along Z axis guarantees that the far and near features are considered for the estimation process which leads finally to a precise ego-motion of the robot. Third, uniform distribution is granted for the features over the whole image which leads to better estimation performance.

Since we are working on a calibrated, rectified and undistorted stereo camera system and we have enough point correspondences, then we can estimate the 3-dimensions of each features point correspondence via triangulation[15]. The re-projection is done using camera matrix 5. Assuming X is a 3D point and P_l and P_r maps the point from R^3 \rightarrow R^2 such as x_l \in R^2 and x_r \in R^2. Error minimization is done using Gauss-Newton optimization iteratively.

\[
\sum_{i=1}^{N} ||(x_l - x_{p_l})||^2 + ||(x_r - x_{p_r})||^2
\]

(4)

Where x_{p_l} and x_{p_r} are the projected value of X on current left and right image planes respectively via projection matrix 5, and x_r and x_l denote the feature locations in the current left and right images respectively. According to [15], ten iterations are sufficient for convergence but in practical a couple of iterations (e.g., 4-8) are enough. To be robust against outliers a RANSAC scheme is used for 50 times to estimate (r,t) using 3 randomly drawn correspondences. Then, the accepted inliers of are used for refining the (r,t) parameters giving final estimated pose.

\[
\begin{bmatrix}
    u \\
    v \\
    w
\end{bmatrix} =
\begin{bmatrix}
    k_x f & 0 & c_u \\
    0 & k_y f & c_v \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    r_{11} & r_{12} & r_{13} & t_1 \\
    r_{21} & r_{22} & r_{23} & t_2 \\
    r_{31} & r_{32} & r_{33} & t_3
\end{bmatrix}
\begin{bmatrix}
    X \\
    Y \\
    Z
\end{bmatrix}
\]

(5)

Last layer is standard Kalman filter. The robot speed is calculated as

\[
v = (rt)^T / \Delta t
\]

(6)

Where (rt)^T is the transformation parameters and \(\Delta t\) is time between frames. Assuming the robot is moving in constant acceleration, we denote acceleration by \(\alpha\). The state equation is given by

\[
\begin{pmatrix}
    v^T \\
    \alpha
\end{pmatrix}^{(t)} = \begin{pmatrix}
    I & \Delta t I \\
    0 & I
\end{pmatrix} \begin{pmatrix}
    v^T \\
    \alpha
\end{pmatrix}^{(t-1)} + \epsilon
\]

(7)

Where I is the 6 by 6 identity matrix. The output equation reduces to

\[
\frac{1}{\Delta t} \begin{pmatrix}
    r^T \\
    t
\end{pmatrix}^{(t)} = \begin{pmatrix}
    I & 0
\end{pmatrix} \begin{pmatrix}
    v^T \\
    \alpha
\end{pmatrix}^{(t)} + \varphi
\]

(8)

Where \(\epsilon, \varphi\) represent Gaussian process and measurement noise, respectively. At the end, the transformation (rotation and translation) between each consecutive stereo pairs is known.

C. Building Depth Map

The input for this step is a rectified and undistorted stereo pair, and the output is a grayscale image represents the depth for each pixel. ELSA (Efficient LArge-scale Stereo) [8] is a new library works with binocular high resolution stereo images with high speed performance. It depends on finding robust matches, then tries to build a prior on the disparities using triangulation on set of support points which are found before (in features extraction step). This strategy reduces the false matching of the remaining points. The resulted dense disparity map leads to an accurate reconstruction. The algorithm is automatically adjust it’s parameters such as searching range and windows size which make it adaptable for environment conditions changing.

D. 3D Reconstruction

Thanks to Point Cloud Library [11] the reconstruction is done very easy since we have the 3D for each point. The X,Y,Z are yielded from camera intrinsic parameters as the following equations show

\[
X = (u - cu) * T / d
\]

\[
Y = (v - cv) * T / d
\]

\[
Z = f * base / d
\]

(9)
where $X, Y, Z$ are the coordinate of 3D point $P$ in space, $u, v$ is a 2D point in the image coordinate system, $cu, cv$ is the principal point of the camera, $d$ is disparity value in $u, v$ and $T$ is the base line which is calculated using

$$T = -P2(1, 4)/P2(1, 1)$$

(10)

Where $P_2$ is the projection matrix of right camera resulted from rectification process.

E. Clouds Registration

From the ego-motion step we have pose (translation and rotation) between each two consecutive stereo pair. The registration here simply thanks to the below equation

$$P_{\text{Cloud}} = P_{\text{Cloud}} + C_{\text{Cloud}} * T$$

(11)

$$T = \begin{pmatrix} r_1 & r_2 & r_3 & t_1 \\ r_4 & r_5 & r_6 & t_2 \\ r_7 & r_8 & r_9 & t_3 \end{pmatrix}$$

(12)

where $P_{\text{Cloud}}, P_{\text{Cloud}}$ represents the previous and current point cloud respectively and $T$ represents the transformation.

IV. EXPERIMENTS

A. Ego-motion Experiment

This experiment includes indoor and outdoor camera trajectory.

1) Outdoor with Random Trajectory Using Pointgrey Flea2: We have a calibrated and set of images sequence for outdoor in urban environment. The camera is moving in smooth trajectory. The first outdoor sequence is download from online source [12] with calibration parameters.

2) Outdoor with Camera Rotation around an Object in Space Using Point Grey Bumblebee2: This experiment is done in outdoor environment moving the camera in full rotation motion. For full rotation, a closed loop circle is the optimal case which is our target, but here the resulted circle is near to close circle because of accumulated error in estimation process.

3) Indoor with Rotation around an Object in Space Using Point Grey Bumblebee2: The experiment is done within indoor environment with a full rotation motion trying to reach a closed loop. Here the trajectory is built from 160 stereo pairs. The figure 2 shows the result for ego-motion experiments.

B. Ego-motion Experiment

1) Registration in outdoor environment with random trajectory using Pointgrey Flea2: A set of 10 clouds in outdoor environment are registered based on transformation we already had from ego-motion process. Then, the clouds are concatenated to each other.

2) Registration in outdoor environment with rotation around an object using Point Grey Bumblebee2: Here we are working in outdoor environment with Point Grey Bumblebee2. 10 clouds are registered upon the transformation we yielded from ego-motion step, and registered using registration equation.

3) Registration in indoor environment with rotation around an object in space using Point Grey Bumblebee2: Last experiment is on a set of points clouds token in indoor environment based on transformation we yielded from ego-motion step, and registered using registration equation. The figure 3 shows the result for Registration experiments.

V. CONCLUSIONS AND FUTURE WORKS

In this paper we proposed an approach for estimating the motion of a stereo camera rig mounted on a vehicle with 6DOF based on visual Odometry. The outlier rejection schema used is RANSAC and Kalman filter. A transformation between each two consecutive frame is calculated. The algorithm is adaptive to environment condition changing, even outdoor or indoor ego-motion. The prerequisites are...
to have a calibrated camera, such that the intrinsic are known, and the camera is moving in smooth trajectory, so the input images are a continues sequence. A strong features matching is used to estimate the dense depth map; 3D points cloud are calculated via triangulation. Finally, register point clouds to have a 3D view for the trajectory. The experiment shows that the estimation is very close to correct trajectory in urban environments. In future, we want to increase the accuracy of our system by adding a GPS/INS system to reduce the estimation accumulated errors which will increase the localization accuracy. Also, we intent to handle dynamic objects environment.

VI. ACKNOWLEDGMENTS

First of all, I would like to thanks Allah Almighty - Alhamdulillah- for giving me the strength, courage and opportunity to undertake this wonderful master and empowering me to accomplish this thesis. I would like to thank my supervisors David Fofi and Cedric Demonceaux for their kind guidance and support during the course of my project. And Finally, I hope that this work will return back in benefit to my country Syria …

REFERENCES


Comparison of Parametric and Non-parametric Learning Methods for the Inverse Dynamics Modelling of the Arm of the iCub Humanoid Robot

Chalikonda Prabhu Kumar and Giorgio Metta

Abstract - Acquiring accurate models of dynamical systems is an essential step in many robotic applications for example safe operation in unstructured dynamic environments. Analytical models for robot dynamics often perform sub-optimal in practice due to various non-linearities and difficulty of accurately estimating the dynamic parameters. Machine learning techniques are less sensitive to these problems and therefore they are an interesting alternative for modeling robot dynamics. Non-parametric methods will approximate the function describing the relationship between joint trajectories (i.e. joint positions, velocities and accelerations) and the joint torques of the rigid body dynamic model. The qualitative measure for the prediction is Root Mean Square Error (RMSE) for forces and torques in three dimensions and influence of the data supplied (i.e. in Batch or Incremental) on RMSE error. In parametric models, we aim to identify a small set of significant parameters of the equations whose form is given to the learning method.

1 Introduction

Robotics research are mainly aimed at designing and implementing systems able to perform advanced tasks in unstructured environments, possibly along with human actors. If the F/T sensor is placed along the kinematic chain as proposed in [1], F/T sensor measures both external and internal forces. This solution allows the robot to detect interactions that occur in the surrounding not only on the end-effector, but on the whole arm. To detect this external contribution of forces acting on the robot the internal forces must be known, estimated or modeled. There are several approaches to estimate these internal forces. Commonly, the inverse dynamics [2] are modeled analytically using

$$\tau = D(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q)$$  \hspace{1cm} (1)

Where D is generalized inertia matrix, C is vector of centrifugal forces and Coriolis, and g is gravity term. This formulation has limited applicability in reality due to the presence of non-linearities (i.e. due to friction, elasticity, flexibility & vibrations) and accurate estimation of kinematic and dynamic parameters. Machine learning forms a viable alternative for modeling robot techniques. In the literature, some learning techniques often outperform analytical models of the rigid body dynamics[3][4]. These techniques were proposed under batch learning when the model is trained off-line. Incremental learning methods adapt the model continuously when new sample of data is added, therefore having an advantage over batch methods for sequential learning. However there are many approaches in the literature but some of these methods does not give any theoretical guarantees on timing requirements, some are hard to tune the parameters etc. The approach used for incremental learning is proposed in [5] with an efficient update rule and stable when compared with other methods. Learning algorithms are based on the well-known regularized least squares algorithms for both linear and non-linear models. Estimating the relationship between the inputs (i.e. joint positions, velocities and accelerations) and outputs (joint torques) referred to functional estimation approach.

A dynamic model is characterized by the inertial parameters of the bodies composing the system which, in case of rigid bodies, are: masses, positions of centers of mass and rotational inertias. These parameters can be estimated using arms joint torques and forces along with position, velocity and acceleration of joint[6]. The idea of identifying inertial parameters are used for object recognition, force control and pose estimation. The inverse dynamics of general kinematic tree is calculated using Recursive Newton Euler algorithm[7]. Recursive Newton-Euler steps can prove that relocation relies solely on inertial parameters[9] & inertial parameters space affects the dynamic equation[10]. More general approach to determine the identifiable subspace, with no assumptions on the regressor structure or on kinematic was proposed by Gautier [12].

In this report presenting non-parametric modeling algorithms for both linear and non-linear models, estimating inertial parameters and showing the effect of regularization parameter $\lambda$ in estimating inertial parameters. In section II non-parametric modeling for both linear and non-linear models for batch and incremental learning, and theory for estimating inertial parameters. Section III with experimental results
2 Methodology

2.1 Linear Models:

Linear regression is a statistical procedure when the relationship between the variables can be described as linear. This means predicting the value of a dependent variable from an independent variable. The output $y$ can be expressed as a linear algebraic combination of the input attributes $x_1, x_2, ..., x_n$. There are different set of methods like ordinary least squares, ridge regression that which are intended for regression in which the target value is expected to be a linear combination of the input variables. In mathematical notation, if $y$ is the predicted value

$$y = w_1x_1 + w_2x_2 + ....$$

The whole objective of the training phase is to learn the weights $w_1, w_2, ...$ by minimizing the empirical loss function. Gradient descent is the classical technique for solving this problem with the general idea of adjusting $w_1, w_2, ...$ along the direction of the maximum gradient of the loss function. To avoid overfitting, regularization (L1 and L2 norms) is used to penalize large values of $w_1, w_2, ..., w_n$. For the implementation GURLS [13] package was used. GURLS is a least squares based library for state of the art supervised learning algorithms.

Implementation is easy using GURLS, there are some tasks which are used for specific purpose. Explaining each task will cover both implementation using GURLS and theory behind the non-parametric modeling.

- **a. Splitting Dataset:** This task is explicitly aimed at splitting the data into training and validation sets. In the training phase we present data along with the outputs to train the model. Validation set is used in order to estimate accuracy of our model has been trained in general.

- **b. Parameter Selection:** Regularization refers to a process to solve an ill-posed problem or to prevent overfitting. This plays a key role in the learning pipeline i.e., regularization parameter is chosen by specifying the approach we are interested in. In these there three different approaches for selecting the regularization parameter $\lambda$. Fix lambda which sets the $\lambda$ value to 1. looceprimal, which performs parameter selection when the primal formulation of RLS is used by leave one out approach. The approximated function is trained on all the data except for one point and a prediction is made for that point. Even though this approach is good, but it is very expensive in computational time. hoprimal, which performs parameter selection when the primal formulation of RLS is used by hold out approach. This is the simplest kind of cross validation. The advantage of this method is usually preferable to the residual method and takes no longer to compute.

- **Optimizer:** The objective of optimizer is to minimize the hyper parameters, so the hypothesis/function is close to outputs for our training samples. For most of the regression problems squared error is the reasonable choice. It is interesting to use the L2 norm, because of its properties, for both the loss function and the regularization terms.

$$G(f) = ||f||^2$$ (2)

and

$$L(y, f(x)) = (y - f(x))^2$$ (3)

The constraining $F$ to the set of linear functions of the form

$$f(x) = \langle w, x \rangle$$ (4)

where $w$ is a weight vector. Inserting the squared loss and L2 regularization results in a convex optimization i.e. inverted bell shape doesn’t have any local optima. Thus, guarantees unique solution which is global optima for problem of $w$, for which the objective function is given by

$$J(w, \lambda) = \frac{1}{2}||w||^2 + \frac{1}{\lambda} \sum_{i=1}^{m} (y_i - f(x_i))^2 = \frac{1}{2}||w||^2 + \frac{1}{\lambda}||y - Xw||^2$$ (5)

In equation 5 uses matrix notation by defining an $m \times n$ matrix input samples $X = [x_1, x_2, ..., x_m]^T$ and an $m$-dimensional vector of outputs $y = [y_1, y_2, ..., y_m]^T$. The additional factor $\frac{1}{2}$ is solely for mathematical convenience and does not affect the optimal solution. Setting partial derivative of $J$ with respect to $w$ to zero, such that

$$\partial J/\partial w = w(\lambda I + X^TX) - X^Ty$$

= 0, we obtain the optimal solution given by

$$argmin_w J(w, \lambda) = (\lambda I + X^TX)^{-1} X^Ty$$ (6)

The system of linear equations is well posed if $\lambda > 0$, this guaranteeing a unique optimal solution. This technique, known as Tikhonov regularization, was originally proposed to improve condition of ill-posed linear systems of equations by Tikhonov and Arsenin, 1977 [14]. The main advantage of this formulation is computational nature.

- **Prediction** Once training of data is finished, prediction of output is performed in this task. This computes the predictions of the linear estimator stored (matrix of coefficient vectors of rls estimator) computed with primal formulation of RLS on the samples passed in the input data matrix i.e. test dataset.
- **Performance Measure** The performance is measured by computing Root Mean Square Error between predicted values and actual F/T measurements.

Data can be provided to the learning algorithm in two ways namely: in Batch and Incremental. In batch we will provide all data at once to the learning algorithm on the other hand for incremental they learn from each training samples as it arrives. In reality incremental learning is preferred over batch learning. These are also termed as off-line and online learning.

**a. Batch Learning For Linear Models**: In batch learning all the data is given at once to learning algorithm. After training was done predictions are made for test dataset. For convenience in the report three bar graphs are shown in one. First bar graph is rmse error of forces (units: Newton) between predicted and actual measurement from the F/T sensor, followed by rmse error for torques (units: Newton⋅meter) and computational time (in Seconds) for both training and testing on vertical axis. On horizontal axis number of samples including testing and approach used for parameter selection. From the results Figure 1 we can see the RMSE error is same for all the experiments this because covariance matrix is stable and as discussed earlier matrix is well posed if $\lambda > 0$. However the error will reduce if the number of samples for training are increased.

**b. Incremental Learning For Linear Models**: The sequence for implementation using GURLS is same as batch learning but data supplied to learning algorithm is different. Initially, taking reasonable amount of samples (above 1000) for training and parameter selection, then we will update model by training each sample. In incremental learning we can do retraining after updating the model, for the experiment it is not considered because it is computationally expensive. Figure 2 we can see error is more because of numerical instability in covariance matrix.

2.2 **Non-Linear Models**

In real life problems linear functions are too restrictive, which are often found to be non-linear. To deal with non-linear models, best way is to apply linear algorithms on non-linear problems to map the input features into feature space. With this approach input-output relation will turn to be in linear form in feature space i.e. Hilbert space. To fulfill this we use kernel in this case. Let us consider map function $\phi : \chi \rightarrow \mathcal{H}$, where $\mathcal{H}$ is Hilbert space which is feature space. On substituting $\phi$ in equation 4, we get

$$f(x) = \langle w, \phi(x) \rangle$$  \hspace{1cm} (7)

optimal solution for equation 6 turns to

$$\arg\min_w J(w, \lambda) = (\lambda I + \Phi^T \Phi)^{-1} \Phi^T y$$ \hspace{1cm} (8)

where $\Phi = \phi(x)$. This means the algorithm operates explicitly in the feature space and consequently $w \in \mathcal{H}$. There are different types of non-linear feature mapping also known as basis functions, which are used in practice like Radial Base Functions, Polynomial basis functions. The time complexity of equation 8 is cubic in the number of features and may therefore be computationally not feasible if huge number of basis functions is made as a choice. To overcome this issue we can use kernel trick. In addition to this it allows linear methods to be applied in infinite dimensional feature spaces. Most of the tasks in the non-linear models are same as linear modeling. In non-linear dual formulation will be used instead of primal formulation. The tasks that are used for non-linear models using GURLS are splitting datasets, parameter selection, kernel selection, optimizer, predicting kernel, prediction and computing error. Using representer theorem the optimal solution of equation 8. We can rewrite $w$ as

$$w = \frac{1}{\lambda} \Phi^T (y - \Phi w) = \Phi^T \alpha = \sum_{i=1}^{m} \alpha_i \phi X_i$$ \hspace{1cm} (9)

showing that $w$ can be written as linear combination of the training samples. Following that m-dimensional coefficient vector $\alpha$ is given by

$$\alpha = \frac{1}{\lambda} (y - \Phi w) = (K + \lambda I)^{-1} y$$ \hspace{1cm} (10)
where \( K = \Phi\Phi^T \). To obtain optimal solutions for this dual representation we need \( m \) – dimensional system of linear equations, as opposed to an \( n \) – dimensional system for primal formulation. The above alternative formulation has one of the most attractive features is computationally advantageous in training when \( m < n \). Apart from this, training occurs within the inner product. The matrix \( K \) is called the kernel matrix and is symmetric and positive semi-definite. It is interesting to know which kernel functions should be used. \( K: \chi \times \chi \rightarrow \mathcal{R} \) to an inner product in a feature space \( \mathcal{H} \). The answer is mercer’s condition, because it avoids explicit expressions for non-linear mapping & makes kernel trick powerful. One of the advantage in using representor theorem is solution for the given problem depends on the inputs only through inner product. The function \( K \) is often called a kernel and to be admissible it should behave like inner product. In addition to this, main use of positive semi definite kernel ensures that the optimization problem will be convex and solution is unique. With the given admissible kernel, it is possible to construct a corresponding Hilbert space \( \mathcal{H} \), which is known as Reproducing Kernel Hilbert Space (RKHS).

In fact, it is difficult to say which kernel is best in performance, there are particular kernels that have been shown to perform well on a wide variety of practical learning problems. Most widely used families of kernels are polynomial and RBF kernel. In experiments linear kernel, RBF kernel and approximating Gaussian kernel with random features are used. Even though polynomial and RBF kernel are widely used in machine learning but they are restricted to use in reality because high computational time & unable to deal with large number of input features. Random features are a trick to speed up supervised learning algorithms, so they are able to handle large datasets.

In the literature, Rahimi and Recht (2008a) [19] demonstrated that the RBF kernel and other shift invariant kernels can be approximated to an arbitrary precision using finite dimensional random features mapping. Their approach utilizes Bochner’s theorem, which relates positive definite functions, among which admissible kernel functions, to Fourier transforms using finite Borel measure. In brief, this theorem states that shift-invariant kernel function \( K(x_i, x_j) = K(x_i - x_j) \) can be described as the Fourier transform of a unique measure \( \mu \). Here \( \mu \) is probability density function. Then the kernel can be estimated randomly sampling features according to \( \mu \). Combining both these techniques, finally we get

\[
k(x_1 - x_j) = \int_{\Omega} e^{-i\omega^T(x_1 - x_j)}\mu(d\omega)
\]

\[
e = \mathbb{E}_\omega [z_\omega(x_1)^T z_\omega(x_j)] \text{ where}
\]

\[
z_\omega(x) = [\cos(\omega^T x), \sin(\omega^T x)]^T
\]

(11)

Inner product \( \langle z_\omega(x_1)^T z_\omega(x_j) \rangle \) gives an unbiased estimate of any shift-invariant kernel \( k(x_1, x_j) \), spectral frequency \( \omega \) is drawn according to its corresponding measure \( \mu \). Corresponding probability density function \( \mu \) can be obtained by computing IFT (Inverse Fourier Transform). Probability density function for the isotropic RBF kernel is Gaussian and it suffice to sample \( \omega \sim N(0, 2\gamma I) \). Predicting kernel task compute kernel matrix between training points and testing points. It must use dual formulation. The prediction step Matrix \( K = \Phi\Phi^T \) obtained above can be described component wise as \( K_{ij} = \langle \phi(x_i), \phi(x_j) \rangle \) and the prediction function can be written as

\[
f(x) = \langle w, x \rangle = \langle \Phi^T \alpha, \phi(x) \rangle = \sum_{i=1}^{m} \alpha_i \langle \phi(x_i), \phi(x) \rangle
\]

(12)

From the above equation 12 we can observe that inner product \( \mathcal{H} \) is required to compute the RLS dual solution. Kernel trick exploits this observation by directly specifying a kernel function. This avoids explicit mapping of the input samples into feature space.

\[
K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}}
\]

(13)

In other way computed the predictions of the regressors stored in Gram matrix on the samples passed in the X matrix. Qualitative measurement is same as linear modeling i.e. RMSE error.

a. Batch Learning for Non-Linear Models: The drawbacks of linear and RBF kernel they are infeasible in large number of features, which need more time and memory. To overcome the issues of regular kernel, approximating the Gaussian kernel with random features are used. From results Figure 3, we can make observations i.e. linear and RBF kernel with less number of samples (1457 for training and 352 for testing) need more time to compute regularization parameter and predictions. Approximating gaussian kernel with random features in the last experiment with complete dataset was used instead of small dataset to alleviate advantage of computational time & able to deal with large number of input features.

b. Incremental Learning for Non-Linear Models: Results Figure 2, Incremental learning for Linear models is numerically unstable, so it leads to more error. So now the idea is to make the system stable by making efficient updates when the new sample arrives. The easiest
method to do update in incremental learning an RLS solution is to perform rank updates on the inverse of co-variance matrix \((\lambda I + X^T X)^{-1}\) using sherman-morison formula. But this approach is numerically unstable because this formula is more sensitive to rounding errors. A numerically much more stable alternative is to update cholesky factor \(R\) of the covariance matrix instead, such that \(RR^T = (\lambda I + X^T X)\). This update strategy is known as QR algorithm in the field of adaptive filtering. Rank updates of the cholesky factor can be computed efficiently, whereas the weight vector \(w\) can be obtained using back substitution. Results in Figure 4 shows that sherman-morrison update is less stable when compared with cholesky update.

### 2.3 Parametric Modeling:

Initial parameters estimation is accomplished by collecting sample measurements if kinematic quantities i.e. joint positions, velocities and acceleration and forces acting in robot i.e. external forces and joint torques. The samples measurements of kinematic quantities are then used to compute regressor that linearly related to inertial parameters to the force measurements. Independently from the method used to obtain a measure quantity that depends linearly in inertial parameters, considering all samples it is possible to write a regressor for all the available measurements in single equation

\[
\begin{bmatrix}
Y_1 \\
Y_2 \\
\vdots \\
Y_N \\
\end{bmatrix}
\phi =
\begin{bmatrix}
f_1 \\
f_2 \\
\vdots \\
f_N \\
\end{bmatrix}
\]  

(14)

Also if the collected samples are sufficiently exciting the equation 14 is almost always intermediate in \(\phi\). So infinite least square solutions exist. As only the elements of identifiable subspace can be identified, to avoid numerical issues it is possible to compute directly the base parameters \(\phi_b = B^T \phi\), where \(B\) is a matrix whose columns are a base for \(I_Y\), that can be calculated from an arbitrary structure using Gautier algorithm[12]

\[
\begin{bmatrix}
Y_1 B \\
Y_2 B \\
\vdots \\
Y_N B \\
\end{bmatrix}
\phi_b =
\begin{bmatrix}
Y_b^T \\
Y_b^T \\
\vdots \\
Y_b^T \\
\end{bmatrix}
\phi_b =
\begin{bmatrix}
f_1 \\
f_2 \\
\vdots \\
f_N \\
\end{bmatrix}
\]  

(15)

Using this equations only the base parameters, the projections on a base of identifiable subspace \(I_Y\) can be estimated. A projection on this subspace is sufficient to obtain many quantities of interest in robotics that solely depends on the inertial parameters. The equation 14 can be used to estimate the parameters \(\phi_b\) using least squares presented in non-parametric modeling part. we are using incremental update, so there will be numerical instability to over this issue cholesky update was preferable than sherman-morrison update. An estimation algorithm is derived from Newton-Euler equations, and used the base force sensor measurements and manipulator (i.e. iCub left arm) joint positions, velocities and accelerations. No direct measurement of the arm’s joint torque or force are required. Adaptive filtering algorithm eliminates the need for difficult to measure joint acceleration. From the Figure 5 we can see that there is change in inertial parameters. It is evident that taking the advantage of \(\lambda\) that prevent overfitting to give accurate prediction in non-parametric modeling, we can say that optimal \(\lambda\) yields to more accurate in obtaining inertial parameters than assumed one (i.e. fixlambda : 1 in our case).

### 3 Experimental Results:

Some experiments are performed to improve the results. Increase in crossfolds for holdout approach will lead for better results. Using 5 cross folds the rmse error is reduced by 0.3 times depicted in Figure 6. Increase in the number of random features for incremental learning of non-linear models Sherman-Morrison update is less stable when compare to the Cholesky update is shown in Figure 7.
4 Conclusion:

In non-parametric model approximating of the function that describes the relationship between inputs (i.e. joints position, velocity and accelerations) and outputs (i.e. forces and torques). This was performed in both linear and non-linear models. In real life robots, linear systems are too restrictive, apart from this the covariance matrix is well posed so it performs better in batch learning but leads to high errors in incremental learning due to instabilities in covariance matrix. In most of the applications incremental learning is preferred than batch learning. However, non-linear models provide an efficient solution towards this context. All the family of kernels like RBF, polynomial etc will not perform well because they are restricted with number of input features, needs lots memory and computational time. To alleviate this issue in non-linear models, Gaussian kernel approximation with random features sounds as an alternative solution for non-linear models. Sherman-Morrison update is numerically unstable, to overcome this cholesky update is preferred. It is stable even with increase or decrease in random features, high dimensional data etc. With more number of features we will penalised by computational time.

In parametric learning method our aim is to identify the small set of significant parameters i.e. mass, center of mass and inertia matrix using Gautier algorithm. Due to presence of cad offsets & F/T offsets it not possible to categorize them for results obtained. However, investigated on regularization parameter $\lambda$ effects estimation of inertial parameters. Since we don’t have ground truth we cannot say they are accurate. From theoretical point of view, role of regularization parameter is to prevent overfitting, this will give accurate predictions in output. In addition to this using Newton-Euler algorithm predictions of forces and torques which are almost equal to ground truth, with this evidence we can say that inertial parameters identified are close to true ones.

These techniques are used in robotics for safe operations. However, learning is common in both but context is different. Future work is to obtain the accurate inertial parameters (i.e. mass, center of mass and inertia matrix). Combining both non-parametric and parametric models for robot tracking control.

5 References

Omnidirectional Vision for the REEM Robot Localization

Jeremie Deray deray.jeremie@gmail.com
Universite de Bourgogne, Le2i UMR 5158 CNRS, 12 rue de la fonderie, 71200 Le Creusot, France
PAL Robotics, C/ Pujades 77-79, 08030 Barcelona, Spain

Abstract—Navigation and therefore localization are two of the most fundamental issues in Robotics. This work aims to investigate the benefits of using a minimalist poly-omnidirectional camera rig for the topological localization of a humanoid robot, namely the REEM robot from PAL Robotics. The proposed system is composed of two non-overlapping fisheye cameras for a complete spherical view acquisition. Such configuration allows for a rich description of the robot environment and is therefore well suited for the localization task.

Visual-based topological localization characterizes the map nodes either in terms of global or local features extracted from images. Following the local feature approach, this work presents a comparison of localization performance between our prototype and the currently embedded cameras. Addressing the kidnapped robot issue, the naive SIFT matching method presented demonstrates the effectiveness of the proposed camera system. Furthermore a comparison between regular SIFT and Spherical SIFT both extracted from omnidirectional camera images is performed.

I. INTRODUCTION

In a near future, robots will be part of our common life, not in factories only anymore but also in public places and our homes. However this will happen only if robots are robust enough to be entrusted with simple tasks. Since most of them implicitly imply a process of localization and navigation, both those fields are extensively active research areas. Moreover robots are still a curiosity for the public, especially humanoid robots such as the REEM robot showed in Fig. 1, which aims to assist people.

In Robotics the most common sensors used for localization are laser range-finders [1], [2]. Despite their accuracy, laser range-finders are expensive, cumbersome and give a restricted profile line of the environment. To overcome those problems, visual-based localization methods have been proposed [3], [4], [5], [6]. Cameras have many advantages, they are compact, increasingly cheap and most of all they are more informative. However conventional cameras have a restricted field of view (FoV), which limits the environment description. This limitation can be overpassed be taking several images of the same location [3] or by defining an acquisition requirement [7].

Omnidirectional cameras are well suited for localization due to their wide field of view and therefore received an increasing attention. However their study is recent compared to conventional cameras. Based on a stereographic projection, the spherical model allows for a intrinsic [8] and extrinsic [9] calibration of an omnidirectional camera rig.

Autonomous Robotics requiring compact and energy efficient sensors a minimalist hardware configuration is proposed in the work. Whereas the rig is composed of 2 only non-overlapping fisheye cameras, it is able to acquire a complete spherical view leading to a rich information on the robot environment.

This work aims at evaluating the benefits of integrating the above mentioned camera system for the topological localization of a humanoid robot, the REEM robot. The growing interest for the use of omnidirectional camera in Robotics localization led to a flourishing literature of the topic. Topological maps aim at sampling the topological space into differentiable subspaces called nodes. Nodes describe places based on their characteristics, in this work their visual characteristics, or visual features. Either global [10], [11], [12] or local [13], [14], [3] features are then used to solve an image classification problem. Assuming a topological map as an image atlas, topological localization is the process of matching the current observation to one of the known environment observations memorized in the atlas.

However, the size of such atlas increases rapidly with the size of the map. It affects then the speed of the matching process. Nodes must be organised in an efficient manner in order to limit the time consumption of the localization. Methods defining connections a node to another [10], [15], [16] aim at focusing the search on a part of the topological map only. They benefit from prior knowledge about the robot localization, assuming that the robot can not move from an extremity of the map to another without crossing the nodes in between. Another approach is to use an efficient and compact representation of the map such as a vocabulary tree [17],

Fig. 1: The REEM robot
The vocabulary tree aims at clustering the dataset by a recursive \( k \)-means classification. Rather than memorizing all the place feature sets, only the tree is memorized whose leaves define the classes (the places).

Currently the REEM robot localization is performed using two laser range-finders together with a Monte-Carlo localization algorithm [18]. Although those sensors are accurate and lead to a correct localization in static environment, their acquired data can be easily corrupted. A new element in the environment or people surrounding the robot may prevent it to localize or lead to incorrect localization. Moreover it embeds 3 conventional cameras for its visual tasks (face recognition, object tracking etc). Those facts led to a study for alternative solutions, more especially the study of localization using omnidirectional cameras.

This work aims at investigating the benefits in the use of omnidirectional cameras for the REEM robot localization. A poly-omnidirectional sensor will be especially prototyped for this work and its particular requirements in terms of environment acquisition and minimal footprint. An evaluation of the camera effectiveness will be conducted and compared to the cameras currently embedded on the robot.

Section II gives references for the reader to consider in details omnidirectional cameras and non overlapping camera rig calibration. Section III details the method for topological localization. Section IV presents the experiments and results. Finally Section V concludes and proposes further work.

II. Minimalist Poly-Omnidirectional Camera Rig Without Overlapping

Unlike conventional cameras and their limited field of view, omnidirectional cameras offer the possibility to acquire globally a scene thanks to their wider FoV, which can be greater than 180 for some fisheye lenses. However one may want to acquire the whole panoramic view of its environment (360 in longitude and 180 in latitude). We briefly introduce in this section the unit sphere model [19], [8] and the extrinsic calibration of non-overlapping camera system, prior to detail the proposed system.

A. The Single Viewpoint Constraint

The single viewpoint constraint is the basis of the unit sphere model. This constraint implies that each ray of light coming to the system will intersect at a single point in space, the focal point. It has been detailed by Baker & Nayar [20]. A camera respecting this constraint is then called central camera.

Moreover Ying & Hu [21] show that fisheye lenses can be considered as central dioptric cameras.

B. Projection Model

Mei [8] proposes a unified model for central omnidirectional cameras, mainly based on the model proposed before by Geyer & Daniilidis [19]. The latter have introduced the idea of using a generic unit sphere \( S^2 \) instead of mirror curves [20].

Moreover Mei has released a Matlab toolbox [22] for the calibration of omnidirectional cameras.

C. Non-Overlapping Cameras Calibration

Similarly to conventional cameras, omnidirectional cameras can be coupled one to another in order to complete the FoV onto the unit sphere (i.e. the ladybug camera from Point Grey). They can be classified into two groups: systems with overlaps of the image from one camera to another and systems without overlaps. Only the latter is considered in this section.

As mentioned in Section II-A an omnidirectional system can be modelled by the unit sphere model if it respects the single viewpoint constraint. However a system made of several cameras cannot physically respect this constraint, each camera has it own unique viewpoint. The generalized camera model of [23] aims at modelling exactly a multi-camera system relying on light path and Plucker vectors. However the main option to overcome this issue is to neglect the baseline between cameras (assuming it as small as possible) and therefore to assume they share a single optical center. This approximation is valid if the distance from a scene point to the camera is very large compared to the distance between cameras optical center [24].

While stereo-system calibration is a common task for systems with overlapping FoV it becomes more challenging for systems without overlap. Several methods have been proposed to this problem. A solution consists of using a mirror such as [25] or [26] so that cameras do share a FoV. However we preferred a simpler method in terms of calibration procedure [9]. Considering a multi-camera system with rigid transformation one to another, the goal is to retrieve the pose of each camera with respect to their respective calibration pattern. By moving the camera rig between the calibration patterns, each camera trajectory \( T_i \) is expressed with respect to their reference frame \( S_i \). The calibration is then reduced to the resolution of two linear systems to retrieve the unknown transformation \( \Delta T \) from a camera \( C_i \) to an other (\textit{slave to master}). Considering only the general motions case of [9] the calibration solution has the advantages to be simple to set up and can be implemented in the calibration scheme of [22]. Thus with a single procedure both intrinsic and extrinsic calibrations are performed.

The reader can refer to [9] for a complete mathematical description.

Once the rotation and translation have been estimated, they can be used to initialize a non-linear refinement to minimize the following cost function:

\[
\sum_i \left\| T_{i}^{-1} \Delta T_i - \Delta T_i T_{i}^{-1} \right\|
\]  

D. Fisheye Camera Rig

The proposed system aims at being integrated onto the REEM robot as showed in Fig. 4 and must then be as embeddable as possible while acquiring as much of the environment as possible. The camera rig is composed of two UI-3240CP uEye cameras from IDS-Imaging on which are mounted fisheye lenses. Cameras are aligned regarding their optical axis with a relative rotation of \( \approx 180^\circ \) over the \( Y \) axis.
Each camera has a full resolution of 1280x1024 however once the fisheye image borders have been subtracted only 1.3M pixels remain for the whole system.

The calibration of our prototype gave us the following result:

$$\Delta R = \begin{bmatrix} -0.9998 & -0.0207 & -0.0026 \\ -0.0209 & 0.9862 & 0.1642 \\ -0.0008 & 0.1642 & -0.9864 \end{bmatrix}$$  \hspace{1cm} (2)

The estimation of $\Delta R$ matches the expected result, that is to say for a pure rotation over $Y$ axis of $\pi$ radian:

$$\Delta R = \begin{bmatrix} -1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & -1.0 \end{bmatrix}$$  \hspace{1cm} (3)

Once the system is calibrated, the individual $S^2$ spheres can be merged into a single one. It implies that the images acquired by each camera can be treated as a single one. Considering the spheres center lying on a same point (translation neglected), the rotation $\Delta R$ is applied to the slave camera. It stitches both hemispherical images as shown in Fig. 2.

![Fig. 2: Two hemispherical images stitched together](image)

To validate the relevance of the calibration, we visually compared the images alignment using the estimated $\Delta R$ and rotation of $\pi$ over $Y$ axis.

![Fig. 3: Zoom of the spherical images](image)

The estimation of $\Delta R$ is applied to the slave camera. It allows for compensating slight misalignments of the cameras.

### III. Topological Localization

#### A. Mapping

The map construction process is performed offline. An operator drives the REEM robot over the place while the robot records an image sequence. Once the place has been fully visited, the map $TM$ is build. Currently we create a node at each frame (1 frame per second –fps) in average every 42.5 cm) of the recorded sequence so that for a sequence of $l$ frames $TM$ holds $l$ nodes. Following a local features approach we characterize each nodes $N_j$ $j \in [1...l]$ as a set of SIFT [27] points $S(P_j)$ extracted from the image pair $P_j$. Moreover the same manner as [13] we benefit from the localization of the robot (SLAM using laser range-finders) to label the nodes with it $L[N_j]$. By doing so we can retrieve a metric estimation of the robot position during the localization phase. The mapping is summarized in Algorithm 1.

#### Algorithm 1 Mapping

**Require**: Cameras video stream

1. while $P_{newImage}$ do 
   2. $P_{newImage} \rightarrow S(P_{newImage})$
   3. Get $L_{newImage}$ from odometry
   4. Associate $L_{newImage} \rightarrow S(P_{newImage})$

#### B. Naive Localization

For a new pair of query images $P_{query}$, their SIFT points are extracted $S(P_{query})$. In a second place we try to match each feature point of $S(P_{query})$ to those of each node $N_j$ of the topological map. The matching is performed using the Fast Approximate Nearest Neighbours Library (FLANN) [28]. We use the $k$ nearest neighbours method (knn) which returns the $k$ closest features in the database. To assert a matching is correct we evaluate the distance ratio as proposed in [27]. The node $N_m$ which has the highest number of matches $M_{sum}$ is then considered as the potentially current location of the robot. An evaluation of the recognition confidence is performed based on the number of matchings. $M_{sum}$ is compared to a threshold $MT_{threshold}$, empirically defined, below which the recognition is considered not reliable. Therefore the robot is considered at an unknown location. Finally, if the recognition confidence is high enough the label $L_m$ of the matched position $N_m$ is retrieved and considered as the robot position estimation. The localization is summarized in Algorithm 2.

#### IV. Experimental Results

#### A. Camera Systems Comparison

To evaluate the interest of integrating the omnidirectional camera rig to the REEM robot as shown in Fig. 4, we compared it to the cameras currently embedded. We have chosen to use REEM’s left eye camera coupled with its back...
Algorithm 2 Localization

Require: Topological map \( T \) with \( l \) nodes

1: \( \textbf{while } P_{query} \textbf{ do} \)
2: \( P_{query} \rightarrow S\{P_{query}\} \)
3: \( \textbf{for } x = 1 \text{ to } l \textbf{ do} \)
4: \( M_{sum} = \text{matching}(S\{P_{query}\}, S\{P_x\}) \)
5: \( \textbf{if } M_{sum} > \text{prev}M_{sum} \textbf{ then} \)
6: \( \text{prev}M_{sum} = M_{sum} \)
7: \( L_m = L\{N_x\} \)
8: \( \textbf{end if} \)
9: \( \textbf{end for} \)
10: \( \textbf{if } \text{prev}M_{sum} > M\text{Threshold} \textbf{ then} \)
11: \( \text{Robot Localized at } L_m \)
12: \( \textbf{else} \)
13: \( \text{Robot not Localized} \)
14: \( \textbf{end if} \)
15: \( \textbf{end while} \)

camera in order to cover as much FoV as possible. For the experiments we performed three indoor sequences, two of them are similar (seq 1 & 2), whereas for the third sequence the robot explores several rooms. We performed two tests for each of the camera systems (2 tests, 2 camera systems, 4 evaluations). For the first test, seq. 1 is used as the map whereas seq. 2 is used as a test. For the second test, seq. 2 is used as map and seq. 3 as test.

Fig. 4: REEM robot with the omnidirectional camera rig

The localization results for each of the 4 tests are shown in Fig. 5. Moreover Tables IV and V detail the results of the second test for both systems. Details are as follows:

- True Positive (TP) – Correct matching, True Negative (TN) – No matching. There exist no node for the position, False Positive (FP) – Incorrect Matching, False Negative (FN) – No matching. There exist a node for the position, Sensitivity (SE) – Performance of the algorithm to localize correctly the robot given that the position exists in the map. Range from 1 (absolute effectiveness) to 0 (null effectiveness) and Specificity (SP) – Performance of the algorithm to determine correctly that the robot is at an unknown. Range from 0 (null effectiveness) to 1 (absolute effectiveness).

The results shown in Fig. 5 and Tab. I highlight the fact that our system is more effective to localize the robot when its current position exists in the map. Also it has a lower mismatch rate than the REEM cameras.

![Fig. 5: Upper row: tests performed with REEM cameras. Lower row: with our system. Blue squares are the map nodes. Red dots are the true robot positions. Lines show matching with MThreshold = 25](image)

B. SIFT vs. Spherical SIFT

From the results presented in Section IV-A we aimed at refining the image matching of the omnidirectional camera system. Mota-Cruz & Al [29] propose an adaptation of the SIFT feature points for spherical images (SSIFT). Relying on Spherical Fourier Transform their work aims at being a direct and robust adaptation of the so called SIFT feature points [27] for images on \( S^2 \). For further details the reader can refer to [29].

To evaluate the benefits of using SSIFT we re-run the 2 tests detailed in Section IV-A for the omnidirectional cameras only. Prior to extract the features, image pairs must be first projected onto the \( S^2 \) as shown in Fig. 2. To our knowledge there exists only a Matlab implementation of the SSIFT, the computation time to extract the features from a single frame is in average 68.9 seconds. This is obviously not suitable for real-time applications as it is.

The same manner as for the regular SIFT we evaluated different values of \( M\text{Threshold} \) from 0 up to 65, results are presented in VI. Moreover Fig. 6 presents the matching results of the second test. Whereas we were expecting results similar to those in Section IV-A, it appeared during our experiments that the SSIFT-based localization is actually performing less than using the regular SIFT. While comparing both experiments for a sensitivity score equal (SIFT : 0.949, SSIFT : 0.927) it appears that the use of the SSIFT produces
4 times more incorrect localizations than using SIFT (SIFT : 10, SSIFT : 40). While comparing for a specificity score equal (SIFT : 0.933, SSIFT : 0.937) the SSIFT is not able to find a correct position for 15 query pairs whereas SIFT misses only 2. The results showed in Fig. 6 and Tab. I show that the regular SIFT are more robust than SSIFT despite the fact that we are using omnidirectional cameras.

Fig. 6: Localization results using SSIFT. $MT_{threshold} = 18$

<table>
<thead>
<tr>
<th>Feature</th>
<th>SE</th>
<th>SP</th>
<th>$MT_h$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>1</td>
<td>0.913</td>
<td>7 cm</td>
</tr>
<tr>
<td>SSIFT</td>
<td>0.927</td>
<td>0.722</td>
<td>15</td>
</tr>
</tbody>
</table>

TABLE II: Comparison of features best performances

C. System Effectiveness in Public Place

Since the REEM robot aims at assisting people in public places and to ensure the results presented in Section IV-A we performed a test simulating the conditions the robot can encounter.

We recorded a fourth sequence during which groups of people are roaming around the robot while it is navigating over several rooms similarly to the third sequence. Moreover several elements of the environment have been moved from a place to another or even replaced. Using the third sequence as the reference map we evaluate in this section the ability of our camera system to localize in a dynamic environment. As previously the robot position is evaluated at each frame, which as been increased to 5 fps (510 images). In average an evaluation is performed every 13.7 cm.

As we were expecting the localization results slightly decreased for this experiment. Fig. 7 shows the general localization matching with 3 closer views. We detail in Tab. VII the localization results for different $MT_{threshold}$ values. For this experiment the best result obtained happened for $MT_{threshold} = 35$ with sensitivity $= 0.739$, specificity $= 0.738$. It is therefore less effective to match the current position to its referent one in the map and to filter out the unknown positions. We recall in Tab. III the results and compare them to those from Tab. V.

Fig. 7: Localization results for the fourth sequence. $MT_{threshold} = 25$

<table>
<thead>
<tr>
<th>Type</th>
<th>SE</th>
<th>SP</th>
<th>$MT_h$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empty office</td>
<td>1</td>
<td>0.913</td>
<td>25</td>
</tr>
<tr>
<td>Busy office</td>
<td>0.739</td>
<td>0.738</td>
<td>35</td>
</tr>
</tbody>
</table>

TABLE III: Localization performances in public place

V. CONCLUSION

We have shown in this work the feasibility of a minimalist poly-omnidirectional camera system composed of only two non-overlapping fisheye cameras. However, due to slight cameras synchronization issue and difference in illumination received by each camera, remains a strong edge at the stitching area. Since traditional histogram equalization methods do not smooth the illumination enough over the two images, this problem requires to investigate methods such as gain compensation, multi-band blending etc [30].

Moreover we demonstrated the benefits of integrating the omnidirectional camera prototype to the REEM robot. For localization task it performs ≈ 50% better in known places and ≈ 51% better to label unknown positions correctly than the currently embedded cameras. The localization method used in this work is said naive because it does not benefit from prior knowledge by exploiting the connections of the map’s nodes. Fig. 7 highlights the fact that many wrong matches happen for two nodes which are far apart one to another. Our current work aims at benefiting from those connections and prior knowledge.

REFERENCES

TABLE IV: REEM cameras. Third test results for different MThreshold

<table>
<thead>
<tr>
<th>MThreshold</th>
<th>TP</th>
<th>TN</th>
<th>FP</th>
<th>FN</th>
<th>SE</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>39</td>
<td>0</td>
<td>149</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>39</td>
<td>1</td>
<td>145</td>
<td>0</td>
<td>1</td>
<td>0.007</td>
</tr>
<tr>
<td>10</td>
<td>39</td>
<td>14</td>
<td>133</td>
<td>0</td>
<td>1</td>
<td>0.094</td>
</tr>
<tr>
<td>15</td>
<td>39</td>
<td>39</td>
<td>90</td>
<td>0</td>
<td>1</td>
<td>0.396</td>
</tr>
<tr>
<td>20</td>
<td>39</td>
<td>105</td>
<td>44</td>
<td>0</td>
<td>1</td>
<td>0.705</td>
</tr>
<tr>
<td>25</td>
<td>39</td>
<td>136</td>
<td>13</td>
<td>0</td>
<td>1</td>
<td>0.913</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>139</td>
<td>10</td>
<td>2</td>
<td>0.949</td>
<td>0.935</td>
</tr>
<tr>
<td>35</td>
<td>34</td>
<td>142</td>
<td>4</td>
<td>8</td>
<td>0.809</td>
<td>0.973</td>
</tr>
<tr>
<td>40</td>
<td>27</td>
<td>143</td>
<td>1</td>
<td>17</td>
<td>0.614</td>
<td>0.995</td>
</tr>
<tr>
<td>45</td>
<td>26</td>
<td>140</td>
<td>0</td>
<td>39</td>
<td>0.133</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE V: Omnidirectional cameras. Second test results for different MThreshold

<table>
<thead>
<tr>
<th>MThreshold</th>
<th>TP</th>
<th>TN</th>
<th>FP</th>
<th>FN</th>
<th>SE</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>39</td>
<td>0</td>
<td>146</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>39</td>
<td>1</td>
<td>145</td>
<td>0</td>
<td>1</td>
<td>0.007</td>
</tr>
<tr>
<td>10</td>
<td>39</td>
<td>48</td>
<td>98</td>
<td>0</td>
<td>1</td>
<td>0.329</td>
</tr>
<tr>
<td>15</td>
<td>38</td>
<td>104</td>
<td>40</td>
<td>3</td>
<td>0.927</td>
<td>0.722</td>
</tr>
<tr>
<td>20</td>
<td>28</td>
<td>133</td>
<td>9</td>
<td>15</td>
<td>0.361</td>
<td>0.937</td>
</tr>
<tr>
<td>25</td>
<td>24</td>
<td>139</td>
<td>1</td>
<td>21</td>
<td>0.533</td>
<td>0.993</td>
</tr>
<tr>
<td>30</td>
<td>14</td>
<td>140</td>
<td>0</td>
<td>31</td>
<td>0.311</td>
<td>1</td>
</tr>
<tr>
<td>35</td>
<td>8</td>
<td>140</td>
<td>0</td>
<td>37</td>
<td>0.178</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>6</td>
<td>140</td>
<td>0</td>
<td>39</td>
<td>0.133</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE VI: SSIFT features. Second test results for different MThreshold

<table>
<thead>
<tr>
<th>MThreshold</th>
<th>TP</th>
<th>TN</th>
<th>FP</th>
<th>FN</th>
<th>SE</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>429</td>
<td>0</td>
<td>81</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>429</td>
<td>0</td>
<td>81</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>429</td>
<td>0</td>
<td>81</td>
<td>0</td>
<td>1</td>
<td>0.998</td>
</tr>
<tr>
<td>15</td>
<td>420</td>
<td>6</td>
<td>75</td>
<td>9</td>
<td>0.980</td>
<td>0.074</td>
</tr>
<tr>
<td>20</td>
<td>396</td>
<td>20</td>
<td>57</td>
<td>37</td>
<td>0.914</td>
<td>0.260</td>
</tr>
<tr>
<td>25</td>
<td>329</td>
<td>48</td>
<td>17</td>
<td>116</td>
<td>0.739</td>
<td>0.738</td>
</tr>
<tr>
<td>30</td>
<td>263</td>
<td>59</td>
<td>0</td>
<td>188</td>
<td>0.583</td>
<td>1</td>
</tr>
<tr>
<td>35</td>
<td>137</td>
<td>59</td>
<td>0</td>
<td>314</td>
<td>0.304</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE VII: Public place. Results of the third test for different values of MThreshold
REAL-TIME 3D HUMAN POSE ESTIMATION FROM POINT CLOUD

Jilliam María Díaz Barros\textsuperscript{1,2}

Le2i - UMR CNRS 6306, Université de Bourgogne, Le Creusot, FR\textsuperscript{1}
Interdisciplinary Centre for Security, Reliability and Trust (SNT), Luxembourg, LU\textsuperscript{2}

ABSTRACT
This paper presents a novel approach to estimate the human pose from a body-scanned point cloud. To do so, a predefined skeleton model is first initialized according to both the skeleton base point and its torso limb obtained by Principal Component Analysis (PCA). Then, the body parts are iteratively clustered and the skeleton limb fitting is performed, based on Expectation Maximization (EM). The human pose is given by the location of each skeletal node in the fitted skeleton model. Experimental results show the ability of the method to estimate the human pose from multiple point cloud video sequences representing the external surface of a scanned human body; being robust, precise and handling large portions of missing data due to occlusions, acquisition hindrances or registration inaccuracies.

Index Terms— Human pose estimation, point cloud, skeleton model

1. INTRODUCTION

Human pose estimation is indispensable in very active research areas such as scene understanding, human-computer interaction and action or gesture recognition. Among the vast literature on this fundamental research topic, many authors have considered predefined human models to simplify the pose estimation task when using conventional 2-D cameras. For instance, Ke et al. in [1] retrieve the human pose from a monocular camera, using downhill simplex algorithm to match 2-D feature points to a predefined 3-D human model. Other approaches specifically parameterize the pose in a lower dimensional space, using skeleton models. An example is the work of Li et al. in [2], where the authors estimate the 2-D human pose in a video sequence using a predefined human skeleton model to fit the silhouette of a body shape. Distance Transform (DT) and Principal Component Analysis (PCA) are used to identify the skeleton base point and to initialize the skeleton. Then, they perform an iterative process to cluster the body parts to which they fit the predefined skeleton model. However, a renewed interest has arisen as a side-effect of the recent advances in 3-D sensing technologies. Indeed, recent consumer-accessible depth cameras such as the Kinect or the Xtion Pro Live provide remarkable advantages, such as easily overcoming the background matting problem, i.e., segmenting foreground objects from the scene background. Although recent approaches based on depth cameras provide very promising human pose estimates, most of them are intended for mono-view systems and thus limited to applications in which the user is facing to the camera. Approaches intended for multi-view systems and thus to estimate the human pose from a full body scan, usually extract a curve-skeleton representation of the shape to which the skeleton model can be fitted and hence, estimate the pose. However, these approaches are impractical for applications in which the human pose must be estimated in real-time.

In this paper, the problem of human pose estimation is addressed in the context of 3-D scenes scanned by multi-view systems composed of multiple depth cameras. To enable for real-time applications, prior knowledge such as a predefined human body skeleton model is also incorporated, from which its skeletal joints will define the configuration and thus the pose of the scanned body. The remainder of this paper is organized as follows: Section 2 presents a review of human pose estimation based on depth sensing. In Section 3 a detailed description of the current approach is presented. Section 4 evaluates the proposed approach on both synthetic and real data. Finally, concluding remarks are given in Section 5.

2. RELATED WORK

Related to single-depth-image pose estimation, Ye et al. [3] proposed a pipeline to combine pose detection with pose refinement. To do so, the depth map is used to find a similar pose within a database of prior full-body surface mesh models. Lehment et al. [4] considered 3-D point clouds extracted from depth maps to fit a mesh of a cylinder-based stickman model using Annealing Particle Filters (APF). However, the aforementioned methods require a GPU-based implementation. Shotton et al. [5] introduced two super-real-time approaches to predict the positions of body joints using a large and varied synthetic set of training images. Decision forests and simple depth-invariant image features are implemented. In [6], Zhang et al. considered a multi-view setup

\textsuperscript{1}This work was supported by the National Research Fund, Luxembourg, under the CORE project C11/BM/1204105/FAVE/Ottersten.
with depth cameras to perform human pose estimation and tracking. The method employs APF and partition sampling in point cloud models, handles occlusions and reduces ambiguities. The initial pose is estimated using a coarse-to-fine search paradigm. To the best of our knowledge, this is the only method using a multiple-depth-camera setup for human pose estimation. Curve-skeleton-extraction approaches have been successfully used for different kind of shapes besides human models. Since they preserve the geometry and topological information of the object, they can be implemented in human pose estimation by approximating the underlying skeletal structure [7]. 

A similar approach was proposed by Sam et al. [9] to handle surfaces with boundaries, polygon soups and point clouds. Although both methods are robust against noise and moderate miss of data, they are not optimized for real-time applications. Tagliasacchi et al. presented in [10] a method to extract curve skeletons based on a generalized rotational symmetry axis (ROSA) of an oriented point cloud. A similar approach was proposed by Sam et al. [11] using the antipodes as a reference. In this case, both methods can handle significant missing data, but require parameter tuning. A real-time curve-skeleton extraction method was proposed by Garcia and Ottersten [7] in which, inspired from [11], the skeletal candidates are extracted in the 2-D space and then back-projected to the 3-D space. The algorithm is robust against significant portions of missing data. Limitations are related to occluded body parts and limbs located very close to each other.

3. PROPOSED APPROACH

In the following, a novel approach to estimate the human pose from a body-scanned point cloud $\mathcal{P}$ describing a set of 3-D points $p_i = (x_i, y_i, z_i)$ representing the underlying external surface of a human body is presented. Similarly to [2], the current approach has considered an articulated human skeleton model composed of 15 nodes and 14 edges, presented in Fig. 1 (a). By doing so, the complexity and flexibility of the human body as well as the high dimensionality of the pose space are reduced. The predefined skeleton model represents a simplified version of the geometry and topology of the human skeleton. Although there are subtle differences between people, human body proportions fit within a fairly standard range and thus, prior knowledge can be considered. Indeed, an average person uses to measure 7.5 times the height of his head (including the head). This in turn allowed to initialize the length of each skeleton limb as shown in Fig. 1 (b) [12].

In this work, $\mathcal{P}$ describes any possible body configuration of an upright person. Hence, the height of a person $(7.5 \times hu)$ is given by the difference between the maximum and minimum $z$ coordinates within $\mathcal{P}$. That is, $hu = (z_{max} - z_{min})/7.5$, being the head the highest body part.

The human pose estimation results from the configuration of the skeletal joints after approximating the aforementioned skeleton model. This is achieved with a four-steps framework. First, both the base point of the skeleton model and the torso orientation are extracted. These two parameters allow the initialization of the torso whereas the remaining skeleton limbs are initialized by an iterative process in which the best initial skeleton limb configuration is selected. Next step concerns the clustering of the body parts, to which finally, their respective skeleton limbs are progressively approximated. The clustering and fitting are performed under a framework based on the theory of Expectation Maximization. Note that the 3-D point clustering to find the torso orientation (Section 3.1) and the initialization of the skeleton model using a predefined set of limb configurations (Section 3.2) are solely performed for the first frame. Indeed, the resulting fitted skeleton corresponds to the initial skeleton for the consecutive frame. By doing so, the time consumed during both stages is reduced, ensuring a better initial skeleton estimate for the following frames.

3.1. Torso and base point extraction

The node $A = (Ax, Ay, Az)$ in Fig. 1 (a) corresponds to the base point of the skeleton model whereas the segment between nodes A and B, skeleton limb $l_1 = B - A$, to the torso. The direction of the torso results from the Principal Component Analysis (PCA) on $\mathcal{P}$. Indeed, the direction of the principal component $v = (vx, vy, vz)$ coincides with the direction of the torso, assuming that $\mathcal{P}$ describes an upright person. The equation of the 3-D line in which the base point $A$ lies is thus defined from the centroid of $\mathcal{P}$, i.e., $\bar{p} = \frac{1}{|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} p_i$, $\forall p_i \in \mathcal{P}$ and the normalized vector $v$. The third coordinate of $A$, i.e., $Az$, is retrieved from the human body proportions denoted in Fig. 1 (b), whereas $Ax = v_x \cdot t + \bar{p}_x$ and $Ay = v_y \cdot t + \bar{p}_y$, with $t = (Az - \bar{p}_z)/(v_z - \bar{p}_z)$. Note that the initialization of the skeleton model and hence, the
body clustering, are strongly dependent on these two parameters. Indeed, a wrong direction of the torso entails to a wrong initialization of the model and thus to an erroneous pose in space. To increase the accuracy and robustness of the torso direction, only those 3-D points that belong to the torso are considered. To do so, the torso 3-D points are classified by fitting a cylinder, a simplified geometric model that can be quickly fitted to the dataset using Random Sample Consensus (RANSAC) [13]. Alternative fitting algorithms with embedded heuristic hypotheses generators can be also considered.

### 3.2. Initialization of the skeleton model

The initial skeleton model results from the skeleton limb’s configuration that minimizes the distances between the 3-D points and the set of skeleton limbs, i.e., the best matching between the predefined skeleton model configuration and the given point cloud \( P \). However, in contrast to alternative approaches to estimate the human pose from 2-D images [2], this task is far from trivial when considering the additional degree of freedom in a 3-D space. First, the skeleton model is aligned to the estimated base point and torso directions. Then, the locations of the remaining skeleton nodes are progressively computed from the set of skeleton limb configurations, presented in Fig. 2, and using incorporated prior knowledge such as the radii of the skeleton limbs and the initial angles of the skeleton joints. Fig. 2 only shows the selected configurations to initialize the right body side. Nevertheless, as can be inferred, the mirrored versions correspond to the configurations of the left body side. Given a skeleton limb \( I_i \) with radius \( r_i \), azimuthal angle \( \theta_i \), and polar angle \( \phi_i \), the 3-D coordinates of the end node \( w = (w_x, w_y, w_z) \) result from:

\[
\begin{align*}
    w_x &= u_x + r_i \cdot \cos \theta_i \cdot \cos \phi_i, \\
    w_y &= u_y + r_i \cdot \sin \theta_i \cdot \cos \phi_i, \\
    w_z &= u_z + r_i \cdot \sin \phi_i,
\end{align*}
\]

with \( u = (u_x, u_y, u_z) \) the 3-D coordinates of the initial node. The spherical coordinate system and the right-hand rule are used to define the initial angles that generate the coordinates of each skeleton node. The angles \( \theta \) and \( \phi \) are fixed within the range of \([0, \pi]\) and \([0, 2\pi]\), respectively. First, the location of the skeleton nodes that are directly connected to the base point \( A \) are computed, i.e., \( B, C, D \) and \( E \). From them, the location of \( F, G, H \) and \( I \), followed by \( J, K, L \) and \( M \) are computed. Finally, nodes \( N \) and \( O \) are computed.

When considering 3-D models that are differently oriented with respect to the \( z \)-axis, the 2\textsuperscript{nd} principal component is used to rotate the initialized skeleton model. In the current system configuration, the eigenvector of the 1\textsuperscript{st} component corresponds to the \( z \)-axis whereas the eigenvector of the 2\textsuperscript{nd} one corresponds to the \( x \)-axis. Thereby, a \( 3 \times 3 \) rotation matrix \( R \) is computed by rearranging the eigenvectors obtained from PCA. The new location of the skeleton nodes is given by \( w = R \cdot u \).

![Fig. 2. Considered skeleton limb and node configurations to initialize the skeleton model (only right body side configurations are shown).](image)

### 3.3. Body parts clustering

After initializing the skeleton model, the clustering of the body parts to which each skeleton limb will be further approximated is performed. To do so, each 3-D point \( p_i \) is assigned to the skeleton limb \( I_k \) to which the distance is minimum, i.e.,

\[
    I_k = \arg \min_{k \in [1,14]} (d(p_i, I_k)) \forall p_i \in P.
\]

Nevertheless, close distances between a 3-D point and two or more skeleton limbs may induce to ambiguity in the clustering. Hence, these 3-D points are not considered within the clustering process.

### 3.4. Skeleton limb fitting

Next step concerns the medial axis estimation of each clustered body part, to which their corresponding skeleton limb will be fitted. It is important to recall that the nodes of the torso, \( A \) and \( B \), are fixed in this step, and the remaining nodes to be computed are only connected to two limbs. From each cluster, the mean point and the three principal components are extracted using PCA. Note that the length of each skeleton limb is known from Fig. 1 (b). To decide which of the three eigenvectors corresponds to the medial axis, they are stretched from both sides starting from the mean point of the cluster and by half of their known length. As a result, three potential medial axes that are not connected to each other, but oriented towards the directions of the principal components, are obtained. The connectivity between the skeleton limbs is ensured by selecting the medial axis candidates of adjacent clusters with the shortest distances between the end node of the previously fitted limb and the initial node of the limb to be fitted. Fig. 3 details the fitting process between the left hip and left thigh skeleton limbs. Red, blue and green segments are the candidate medial axes of each cluster, depicted as cylinders. The three candidates of each cluster have exactly the same length, which corresponds to the length of their respective skeleton limb. The yellow point in Fig. 3 (a) denotes the end node of the hip limb \( G' \), whereas the orange dashed line is the shortest distance to the skeletal node candidates of the adjacent cluster, i.e., \( G'' \). Next, the centroid \( \bar{q} \) between \( G' \) and \( G'' \) is calculated, shown in purple color in Fig. 3 (b), and the fitting of \( I_6 \) is refined by reorienting it towards \( \bar{q} \), as shown in Fig. 3 (c). The new 3-D coordinates of \( G \) result from \( G = B + \lambda \) with \( \lambda = d(B, G) \cdot (\bar{q} - B)/||\bar{q} - B|| \). Note that the node \( G \) corresponds to the initial node of the
adjacent skeleton limb, *i.e.*, the thigh. Therefore, the skeleton limb $l_{10}$ is translated to its respective location, which gives an initial location for the node $K$, to be refined when fitting its adjacent skeleton limb, *i.e.*, $l_{14}$.

### 3.5. Skeleton refinement

Similarly to [2], the fitting of the skeleton model is performed through an iterative process based on Expectation Maximization (EM). The Expectation (E) step comprises Section 3.3 and the first part of Section 3.4, where the expected human skeleton is calculated for the current pose. The Maximization (M) step corresponds to the last part of Section 3.3, where the parameters that maximize the expectation of the skeleton model during the fitting process are computed. From the experiments, the fitting process converges to a good pose estimation in only one or two iterations.

### 4. EXPERIMENTAL RESULTS

In the following, the proposed human pose estimation approach is evaluated on both real and synthetic data. All reported results have been obtained using a Mobile Intel® QM67 Express Chipset with an integrated graphic card Intel® HD Graphics 3000. The proposed approach has been implemented in C++ language using the OpenCV [14] and PCL [15] libraries. Real data has been recorded using a multi-view sensing system composed by 2 consumer-accessible RGB-D cameras, *i.e.*, the Asus Xtion Pro Live camera, with opposed field-of-views, *i.e.*, with no data overlapping. Nevertheless, the relationship between the two cameras was determined by a calibration step, using the stereo calibration implementation available in OpenCV [14]. Better registration approaches based on ICP, bundle adjustment or the combination of both can also be considered. However, the current approach perfectly estimates the human pose on such a coarse registered point clouds, handling large portions of missing data as well as registration inaccuracies. Synthetic data has been generated using V-Rep [16], a very versatile robot simulator tool in which the user can replicate real scenarios. A simulated scene has been created to generate the test cases with four virtual Kinect cameras installed in the top corners of the virtual scene. It can be observed that the data is perfectly registered since a full knowledge of the relationship between the cameras and their calibration parameters is known. Consequently, synthetic data has been considered as ground truth data in the next evaluations. All datasets have been voxelized to account for point redundancy after data registration. Voxelization stands for a discrete approximation of 3-D objects into a volumetric representation [17].

Some visual results on both synthetic and real body-scanned datasets are shown in the Fig. 4. The first row presents the considered datasets highlighting the estimated orientation of the torso (green line) and the extracted base point (purple dot). Second row shows the initialization of the predefined skeleton model. The clustering of all body parts is shown in the third row whereas the fitted skeleton model is shown in last row, from which results the human pose. In Fig. 5, the estimated poses of the synthetic Bill model and a real dataset on some selected point-cloud video frames are shown. These results show that the method is able to accurately estimate the body pose of different body configurations of an upright person.

#### 4.1. Robustness to noise

Next, the robustness of the current approach against noise is evaluated. To do so, the synthetic dataset of Bill has been considered, to which it has been applied a zero-mean Gaussian noise with standard deviation $\sigma \in [0.5, 5]$ cm, *i.e.*, $\mathcal{N}(0, \sigma^2)$. In order to increase the reliability of this evaluation, the noise has been added to the depth maps acquired by each virtual Kinect camera, *i.e.*, before being transformed to point clouds. Fig. 6 depicts the error between the location of the resulting skeleton nodes from the noise-free 3-D model (considered as ground truth), and their respective ones from the noisy models. As shown in Fig. 7, the proposed approach is able to estimate the Bill pose for all $\sigma$ values. However, it can be observed that the fitting error of the skeletal nodes corresponding to body extremities slightly increases with amount of noise.

#### 4.2. Runtime and performance analysis

Table 1 reports the time consumption to estimate the human pose for each of the datasets presented in Fig. 4. Note that most of the time is dedicated to cluster the 3-D points to estimate the direction of the torso. However, it is important to recall that this operation might be done only once in the first video frame, as discussed in Section 3.

If a better performance is required, one can increase the voxel size to represent the body-scanned point cloud. Thus, a voxel size of 1 cm$^3$ has been considered when evaluating...
Fig. 4. Human pose estimation on body-scanned point clouds using both real and synthetic data. 1st row, input point cloud. 2nd row, initial skeleton model. 3rd row, body parts clustering. 4th row, approximated skeleton model. 1st col., Nilin Combat dataset. 2nd col., Iron Man dataset. 3rd col., Jilliam dataset. 4th col., Frederic dataset.

Fig. 5. Human pose estimation on different frames from synthetic (1st and 2nd rows) and real (3rd row) video sequences of body-scanned point clouds.

Fig. 6. Fitting error of skeletal nodes for $\sigma \in [0.5, 5]$ cm.

Fig. 7. Bill pose estimation for (a) $\sigma = 0$ cm, (b) $\sigma = 2.5$ cm and (c) $\sigma = 5$ cm.
performed in parallel.

and left and right legs, that are independent and thus, can be

native techniques to segment the 3-D points belonging to the

even in the presence of high noise within the depth measure-

estimate is achieved in both synthetic and real datasets, and

imization. From the experiments, it is shown that a good

an iterative process based on the theory of Expectation Max-

model is initialized and then fitted to the given point cloud by

knowledge on human body proportions, a predefined skeleton

point cloud datasets has been described. Using PCA and prior

A scheme to estimate the human pose from body-scanned

be divided in four different regions,

improvement may be achieved by parallelizing the initializa-

constraint of this implementation is that there has to be a min-

remains accurately estimated, as shown in Table 2. The only

voxel size to represent the Ironman dataset (units are in ms).

Table 2. Time consumption analysis for human pose esti-

mation on the datasets presented in Fig. 4 (units are in ms). Reported are the mean values taken over 100 iterations. 1st row, Nilin Combat dataset (12655 points). 2nd row, Iron Man dataset (21975 points). 3rd row, Jilliam dataset (14855 points). 4th row, Frederic dataset (19675 points).

<table>
<thead>
<tr>
<th>Data set</th>
<th>Cluster 3-D torso points</th>
<th>Torso and base point extraction</th>
<th>Initialize skeleton model</th>
<th>Cluster body parts</th>
<th>Skeleton limb fitting</th>
<th>Total time (1st frame)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nilin Combat</td>
<td>226.2</td>
<td>5.0</td>
<td>238.7</td>
<td>47.8</td>
<td>28.0</td>
<td>545.7</td>
</tr>
<tr>
<td>Iron Man</td>
<td>316.8</td>
<td>8.5</td>
<td>412.4</td>
<td>82.1</td>
<td>47.2</td>
<td>867</td>
</tr>
<tr>
<td>Jilliam</td>
<td>219.3</td>
<td>5.4</td>
<td>274.5</td>
<td>56.2</td>
<td>30.8</td>
<td>586.2</td>
</tr>
<tr>
<td>Frederic</td>
<td>220.2</td>
<td>5.0</td>
<td>275.8</td>
<td>54.7</td>
<td>32.0</td>
<td>587.7</td>
</tr>
</tbody>
</table>

the approach. However, the performance is significantly in-

creased by increasing the voxel size, whereas the human pose

remains accurately estimated, as shown in Table 2. The only

constraint of this implementation is that there has to be a min-

imum distance of 5 cm between two 3-D points to be clustered

within the same object. Indeed, this preliminary clustering is

performed to cluster each individual in the scene. Another

improvement may be achieved by parallelizing the initializa-

tion of the skeleton model, the clustering of body parts, and

the fitting of the skeleton limbs. Indeed, the human body can

be divided in four different regions, i.e., left and right arms

and left and right legs, that are independent and thus, can be

performed in parallel.

5. CONCLUDING REMARKS

A scheme to estimate the human pose from body-scanned

point cloud datasets has been described. Using PCA and prior

knowledge on human body proportions, a predefined skeleton

model is initialized and then fitted to the given point cloud by

an iterative process based on the theory of Expectation Max-

imization. From the experiments, it is shown that a good

estimate is achieved in both synthetic and real datasets, and

even in the presence of high noise within the depth measure-

ments. As future work, the current approach will be extended
to alternative body configurations other than upright. Alter-

native techniques to segment the 3-D points belonging to the
torso (used to estimate its direction) will be further investi-

gated in order to address the limitations given by the cylindri-

cal model, such as the tuning of the model parameters.

6. REFERENCES


SOS Optimization for a special class of minimal problems in Computer Vision: an application to rotation estimation

Deepak Gurung
University of Burgundy

Adlane Habed, Nicolas Padoy
ICube, University of Strasbourg

Cedric Demonceaux
Le2i, University of Burgundy

Abstract

This paper deals with the problem of estimating an unknown parametric model of transformation and data correspondences. We introduce a new algorithmic technique based on geometry and optimization to solve these two problems. The proposed method uses the branch-and-prune (B&Q) paradigm to search over the space of camera transformation parameters. We propose a method that combines B&P and sum-of-squares(SOS) optimization method to certify the global optimality of the search.

Relying on the geometric consistency of feature points, our algorithm outputs the correspondences with largest number of inliers. In our approach, we formulate the transformation estimation problem as an optimization over polynomials. These polynomials belong to a class that can be subjected to SOS optimization. We do so by formulating the problem as a SOS polynomial using Positivstellensatz-based relation and solving it using Linear Matrix Inequalities (LMIs) technique. In this paper, we apply this algorithm to estimate the rotation in a purely-rotating camera. We demonstrate good performance of estimation and data correspondences on the synthetic data.

1. Introduction

Most existing methods deal with Structure and Motion problems in two steps, pairwise correspondences between image pixels in two images are first established before the motion of the camera (and 3D structure if applicable) is calculated. The point correspondence problem can be solved using feature similarity. Such similarity measures are plagued with ambiguities, resulting in mismatched points across two images to be falsely paired as point correspondences. The alternative approach is to rely on geometric consistency. Best results can be obtained by combining both approaches is to combine both of these approaches. In our method, we use the geometric consistency of point correspondences to identify inliers and outliers.

In this paper, we address two problems related to geometric Computer Vision: estimation of a global solution for the geometric transformation and the identification of inliers/outliers. In fact, it turns out that the two problems are related, that is, these two problems can be solved simultaneously. Firstly, in estimating transformations, we turn the geometric problem into one of solving a system of polynomials. In the considered multivariate system of polynomials, we exploit some of the properties of polynomials—namely Positivity and Sum of Squares (SOS)—and together with linear matrix inequalities (LMI), we check whether they are solvable or not. Secondly, for identifying inliers, the basic principle is the commonality in solutions of the polynomials for each point correspondence. This solvability of polynomials and the commonality of their solutions implies geometrical consistency of point correspondences.

We use geometry and appearance feature to identify inliers. For a pair of images, putative correspondences can be obtained by matching well-known feature descriptors like SIFT [7] and SURF [1]. Feature matching methods relying solely on local similarity have ambiguities, for instance when the texture is similar throughout the image. To eliminate such ambiguities, we check the geometry induced by such correspondences. Each of these correspondences are subjected to SOS feasibility test. This test certifies whether the correspondences have any geometric relation. Outliers can be easily identified by their success in SOS feasibility test. Other correspondences that are not SOS feasible may converge to different bounds. Among these bounds, the one that is agreed upon by the highest number of correspondences is the globally optimal bound.

Our algorithm is tested on the rotation estimation problem which involves solving a system of quadratic trivariate polynomial equations. Numerical investigations are carried out on synthetic data in the presence of image noise. The results show that the proposed method provides significantly better estimates for rotation and inliers identification than homography-based methods using RANSAC scheme as well as better minimization of Sampson error and root mean square error.

Related work: Polynomial solving for minimal problems are used within a RANSAC framework to weed out
outliers from noisy point correspondences. For example, consider the polynomial solvers employed in the relative pose estimation problem. This problem includes finding rotation and translation across an image pair. Based on the parametrization of the relative pose problem for two views, the minimal number of point correspondences required is five [5]. There are methods that provide an analytical solution to the polynomial equations [5, 10, 8]. A practical solution to the five point algorithm was given by Philip [10] by extracting the roots of a thirteenth-degree polynomials. As a further improvement of five-point algorithm, Nister [8] derived solutions as the roots of a tenth-order polynomial which is solved using Sturm’s root-bracketing approach. These methods find analytical solutions. They however cannot handle overdetermined polynomial systems.

In addition to analytical solutions there are methods proposed for finding numerical solution of polynomial equations arising in the relative pose problem, for example [12]. The state-of-the-art method for numerically solving this problem in [12] is grounded in the theory of algebraic geometry. The general outline of such algorithms is to calculate the Grobner bases of the geometric problems. It derives the Grobner basis of polynomials on the essential matrix and constructs a matrix called action matrix, whose eigenvalues and eigenvectors provide the solution. In another such method, where at least eight point correspondences are known, Longuet-Higgins [6] provides a solution based on the essential matrix.

Another interesting way of solving polynomials is using Homotopy Continuation method [13]. This method has been applied to solve the structure and motion [11] and camera self-calibration problems [3, 4]. Homotopy continuation is an iterative process that aims at finding all the solutions. It starts with a calculated zeros for an arbitrary polynomial and then gradually deforms the polynomial, solving the original problem at hand. The problem with this method is that the zeros of polynomials can turn real solution into a complex one. In such a case there is an ambiguity introduced by imaginary values for solutions.

Contributions: In particular, the contributions of the present thesis are as follows:

• We devise a globally optimal transformation estimation method based on the branch-and-prune paradigm and SOS optimization technique. Transformation estimation problem in Computer Vision is formulated into an optimization over a system of polynomials.

• We develop a method to maximize the number of inliers. This is possible by considering each of the correspondences separately. Those correspondences that satisfy the geometric and appearance similarity constraint converge to a bound. The highest number of correspondences agreeing on the same bound are identified as inliers.

2. Problem Overview

The main objective of the present thesis is to devise an algorithm for finding the roots of a class of polynomial equations. This class of multivariate polynomials can be decomposed into sums-of-squares polynomials. The resulting algorithm must be able to estimate such transformation and be robust in presence of noisy image correspondences and outliers (defined in section 2.2). In addition the method has to be globally convergent and guarantee the optimality of the solution. In this chapter we provide a mathematical formulation to our problem.

2.1. Polynomials

Let us consider two images, left and right images, which are related by a geometrical transformation. We denote $p_i$ and $p'_j$ the correspondence point in left and right images. Suppose there are $N_l$ and $N_r$ number of features in left and right images. Let $\theta$ be a vector of $n$ unknown transformation parameters. Consider the number of $m$ polynomials defined by each point pair. This transformation is expressed by the polynomials $f_k$ where, $k = 1, \ldots, m$. Our goal is to find $\theta$ such that $f_k(\theta) = 0, \forall k = 1, \ldots, m$. In a noise-free scenario, points that are consistent with the transformation satisfy the polynomial equality $f_k$ are referred to as inliers.

2.2. Inliers and Outliers

Matching points using a feature based method like the SIFT descriptor [7] does not always result in point pairs that have geometrical consistency. Let $a_i$ and $a'_j$ are the features for the points $p_i$ and $p'_j$ respectively. The problem then is to identify inliers that have geometrical consistency (i.e., $|f_k(\theta)| \leq \sigma$) and have similarity in feature, i.e, $h(a_i, a'_j) \leq T_h$, where the distance function $h$ is lower than threshold $T_h$. In this section we provide the representation of inliers and outliers. Suppose $z_{ij}$ refers the correspondence between the point pair $(p_i, p'_j)$. Then,

$$z_{ij} = \begin{cases} 1 & \text{if, } |f_k(\theta)| \leq \sigma \text{ and } h(a_i, a'_j) \leq T_h, \\ 0 & \text{otherwise.} \end{cases}$$

(1)

Inliers are represented as $z_{ij} = 1$ and outliers are represented as $z_{ij} = 0$.

2.3. Problem Statement

In our method we divided the variable space $\theta$ into bounds $[\underline{\theta}, \overline{\theta}]$. We relax fitting a value $\theta$ in the polynomial over constraints to searching a bound $[\underline{\theta}, \overline{\theta}]$ of the parameter that has zero crossing of the polynomial $f_k$. Hence, we can formulate our problem as,
arg max \( z_i \), \( z_j \) \( \sum_{i}^{N_{i}} \sum_{j}^{N_{j}} z_{ij} \)

subject to, 
\[
\begin{align*}
&z_{ij} h(a_i, a_j) \leq z_{ij} T_h, \quad \forall i, j \\
&\exists \theta \in [\theta, \bar{\theta}], \quad z_{ij} f_k(\theta) = 0, \quad \forall i, j, k, \ \\
&\|\bar{\theta} - \theta\| \leq \epsilon, \quad \forall i, j, k, \ \\
&z_{ij} \in \{0, 1\}, \quad \forall i, j.
\end{align*}
\]

The first constraint \( z_{ij} h(a_i, a_j) \leq z_{ij} T_h \) has not changed from the original formulation. This condition is for feature similarity and \( T_h \) is the user-selected threshold. The second constraint establishes the relaxation of \( \theta \) to \([\theta, \bar{\theta}] \). The roots of polynomials \( f_k(\theta) \) must lie within this bound. This constraint along with third constraint \( \|\bar{\theta} - \theta\| \leq \epsilon \) enforces that, for a geometrically consistent point pairs, that are the inliers, the model \( \theta \) fitting the polynomial equality is within user-selected tolerance threshold \( \epsilon \).

The second constraint of equation [2] in particular, imposes the condition that the bound must possess a root of a polynomial. The problem is then to provide proof of existence or nonexistence of a solution \( f(\theta) = 0 \) in the bound under consideration \([\theta, \bar{\theta}] \). From an optimization perspective two related approaches provide this proof. The one is the primal approach and the other is the dual approach. Both methods provide existence/nonexistence of the solution in the bound. The dual approach is based on sum-of-squares (SOS) based optimization.

3. Proposed Sum of Squares optimization

We propose a method that combines B&P and sum-of-squares(SOS) optimization method to certify the global optimality of the search. We branch the space of unknown variables of polynomials into multiple bounds, and subject each of them to some form of test. The guarantee of inexistence of solution within a bound can be provided by checking if the polynomial is SOS feasible within the bound. We use Positivstellensatz-based relation that provides irrefutable evidence of the existence of solution within the bound under consideration. We formulate the problem of polynomial systems as a SOS feasibility problem and solve it via LMIs technique.

3.1. Sum of Squares

A multivariate polynomial \( p(x) \) for \( x \in \mathbb{R}^{n} \) is sum-of-squares if there exist, polynomials \( q_1(x), \ldots, q_m(x) \) such that \( p(x) = \sum_{i=1}^{m} q_i^2(x) \). If polynomial \( p(x) \) is a sum of squares, then it obviously satisfies \( p(x) \geq 0 \) for all \( x \in \mathbb{R}^{n} \). Thus, an SOS condition is a sufficient condition for global nonnegativity. Positivity of \( p(x) \) can be certified by decomposing \( p(x) \) into \( q_i(x) \). This process is called SOS decomposition. D. Hilbert’s 17th problem investigates the positivity of polynomials and the existence of sums of squares (SOS) for such polynomials [9]. The SOS decomposition of polynomial \( p(x) \) is equivalent to the existence of a positive semidefinite matrix \( Q \), and a properly chosen vector of monomials \( z(x) \) such that \( p(x) = z^T(x)Qz(x) \).

3.2. Constrained SOS Feasibility

Sum of squares method can be used to check positivity of polynomials \( f_i(x), i = 1, \ldots, m \) over bound \( q_j(x), j = 1, \ldots, \infty \). By construction, \( q_j(x) \) are positive polynomials. The polynomials \( f_i(x) \) for \( i = 1, \ldots, m \) can be formulated into SOS polynomials. One way to formulate SOS polynomial is via Positivstellensatz-based relaxations, in short P-satz. This method that allows any polynomial to be represented by SOSs can be seen as a generalization of the S-procedure. The basic idea is to show positivity of \( f_i(x) \) search for a sum-of-squares polynomial, \( s_j(x) \), such that \( f_i(x) \geq s_j(x) s_j(x) \). The degree constrained Psatz-based relation including \( f_i(x) \) and \( s_j(x) \) as,

\[
F(x) - G(x) \geq 0,
\]

where \( F(x) = \sum_{i=1}^{m} t_i f_i, \ t_i \) is any polynomial and \( G(x) = \sum_{j=1}^{\infty} s_j g_j \) represents all the polynomials formed by multiplying \( s_j \) with SOS polynomial, \( g_j(x) \). If \( \exists t_i, s_j \) such that equation [3] is satisfied, then \( F(x) \) is SOS decomposable.

Testing a P-satz based relation, equation [3] is a convex feasibility problem in \( t_i, s_j \). To solve it, we consider a subset of the cone to search, i.e., the maximum degree of polynomial \( G(x) \) to be bounded. If degree of \( F(x) \) is 2\( d \), we consider only polynomials with degrees equal to or less than 2\( d \) for \( G(x) \). The infeasibility refutations for large degrees are not necessary as explained in [9].

3.3. Linear Matrix Inequalities (LMIs)

The polynomial equation [2] can be solved for \( t_i, s_j \) using LMI method. To do so, we need to formulate this equation into LMIs form. A linear matrix inequality (LMI) in the variable \( x \in \mathbb{R}^{n} \) has the form

\[
A(x) \preceq A_0 + x_1 A_1 + \cdots + x_n A_n \succeq 0
\]

where \( A_0 \in \mathbb{R}^{k \times k} \), \( \ldots, A_n \in \mathbb{R}^{k \times k} \) are symmetric matrices. \( x = (x_1, \ldots, x_k) \) is a vector of \( n \) real numbers called the decision variables. The LMI is a convex constraint on \( x \), i.e., the set \( \{ x | A(x) \succeq 0 \} \) is convex.

4. The Rotation Problem

The goal is to apply the method we have devised for polynomial solving to estimate the rotation of a purely-rotating camera.
4.1. Camera Model

Let \( M \) be the mapping matrix of perspective camera from 3D projective space \( \mathcal{P}^3 \) to 2D projective space \( \mathcal{P}^2 \). It takes a point \( \mathbf{X} = (x, y, z, 1)^T \) to \( \mathbf{x} \approx M \mathbf{X} \). A camera matrix may be decomposed as \( M \approx K (R - Rt) \), where \( t \) represents the location of camera, \( R \) the rotation matrix of the camera with absolute coordinate frame, \( K \) the upper triangular matrix called calibration matrix of camera. We assume the calibration matrix \( K \) is known and remains constant for all the images. The matrix \( (R - Rt) \) represents a rigid transformation of \( \mathbb{R}^3 \). For a rotation only camera, the camera matrix is \( M \approx K (R(0) = KR) \).

4.2. Two view geometry for rotating camera

Let the two cameras be \( M_1 \approx KR_1 \) and \( M_2 \approx KR_2 \) and the projected image points are \( x_1 \approx KR_1 \mathbf{X} \) and \( x_2 \approx KR_2 \mathbf{X} \). As we already know the calibration matrix \( K \), we rearrange the formalization for the projected points as, \( K^{-1} x_1 \approx R_1 \mathbf{X} \) and \( K^{-1} x_2 \approx R_2 \mathbf{X} \). This gives, \( K^{-1} x_2 \approx R_2 R_1^{-1} K^{-1} x_1 \). For calibrated images, we simply consider \( x_2 \approx R_2 R_1^{-1} x_1 \). Here \( R_1 \) and \( R_2 \) are absolute poses of cameras with respect to the world coordinate. Now, formulating the problem as a relative pose problem, let orientation of image 1 be initial orientation that is \( R_1 = I_{3 \times 3} \) and \( R \) be the rotation of camera between image 1 and 2. Then we have, \( R = R_2 R_1^{-1} = R_2 I_{3 \times 3} = R_2 \). Then,

\[
x_2 \approx R x_1. \tag{5}
\]

In equation \( 5 \) we know both the calibrated image points \( x_1 \) and \( x_2 \), and we need to estimate the unknown parameters of \( R \). Next we introduce parameterization of the Rotation matrix using Cayley’s transformation.

4.3. Rotation parametrization

Rotation can be parametrized by Cayley’s transformation. The advantage of using Cayley’s transformation is that this transformation provides minimal parameters for the parameterizing rotation matrix. Moreover, Cayley’s parametrization provides a symmetric parametrization of Rotation matrix. Let Cayley parameters be represented as \( \mathbf{v} = (v_x, v_y, v_z)^T \). We consider the following Cayley transform for rotation parametrization

\[
R = (I - [\mathbf{v}]_\times)^{-1} (I + [\mathbf{v}]_\times) \tag{6}
\]

where, \( [\mathbf{v}]_\times \) represents a skew-symmetric.

4.4. System of Quadratic Trivariate Polynomials

Using rotation parameters, Eq. \( 6 \) and substituting it in Eq. \( 5 \) we obtain three polynomial equations. By rearranging the terms and taking the cross product of these collinear vectors yields the zero vector. That is,

\[
(I - [\mathbf{v}]_\times) x_2 \times (I + [\mathbf{v}]_\times) x_1 = 0_{3 \times 1}. \tag{7}
\]

On solving the equation above we get a system of polynomials on variable \( v_x, v_y \) and \( v_z \) as,

\[
f_i(v_x, v_y, v_z) = 0, \quad i = 1, 2, 3 \tag{8}
\]

Of these polynomials only the first two polynomials are linearly independent. The last polynomial is a linear combination of the first two polynomials. This implies that each correspondence point gives rise to two independent equations. For three unknown variables at least three such equations are required. Hence a minimum of two correspondence points are required for estimating all three parameters of Cayley’s transformation.

4.5. LMI formulation for purely-rotating camera

Assume the bounds are given as \( B(v_x, v_y, v_z) = \{v_x \leq v_x \leq v_{xu}, v_y \leq v_y \leq v_{yu}, v_z \leq v_z \leq v_{zu}\} \). This bounds defines the polynomials \( G(x) \) of equation \( 3 \). \( G(x) \) for purely-rotating camera is composed of a number of polynomials which is influenced by the degree and number of variables of polynomial \( f_i(.) \) in equation \( 8 \). The detail to obtain LMI formulation is given in \( 2 \). The final for of Psatz based equation turns out to be,

\[
\sum_{i=1}^{3} t_i f_i - \sum_{j=7}^{21} s_j g_j \geq 0 \tag{9}
\]

where \( j = 7, \ldots, 21 \) defines second-order polynomials formed from \( B(v_x, v_y, v_z) \). The linear polynomials (for \( j = 1, \ldots, 6 \)) are discarded. This is because \( s_j \) polynomials can only be of even degree. Applying Gram matrix method on each polynomials of equation \( 9 \) we obtain,

\[
\sum_{i=1}^{3} t_i Q_i - \sum_{i=7}^{21} s_i Q_i \geq 0, \quad s_i \geq 0 \tag{10}
\]

where \( Q(x) \) is a Gram matrix such that \( f(x) = x^T Q x \).

5. Experiments with Synthetic Data

Our algorithm was tested on synthetic data to determine its performance in the presence of noise. The synthetic data consisted of 500 3D points distributed evenly in front of the camera at viewpoint 1. We then randomly rotate the camera to select viewpoint 2 meanwhile ensuring that the rotation has at least 20 points overlapping in both viewpoints for a more realistic purpose. The simulated camera was assumed to satisfy the pinhole model. It took images of size 256 × 256 pixels. The magnification factors \( k_u \) and \( k_v \) both equal to 100. The distortion was assumed to be \( d = 0 \) and the principal point of the image was located at \((u_c, v_c) = (128, 128)\). For each experimental run, 20 sets of points were selected from the entire overlapping point sets of the two viewpoints. Finally, the image coordinates computed using these 20 point sets were used to estimate the rotation.
5.1. Performance in the presence of noise

For each synthetic data set, varying degree of noise was added and rotation estimation was carried out. Gaussian noise varying from 0 to 2.5 pixels was added to image. The algorithm is executed for 1000 runs for all noise level. The results are shown in figure [1]. It summarizes the influence of noise on the accuracy of our algorithm compared to that of RANSAC.

Findings: When all of the inliers are corrupted with Gaussian noise, the results were better for our method compared to that of RANSAC alone. Both algorithms have comparable performance until noise of 1.2 pixels. Beyond this level, our algorithm performs better, both in terms of the root mean square error and Sampson error (figure [1]).

In comparing the number of inliers estimated by our method and by RANSAC, we observed that our method identified more number of inliers than that of RANSAC. This is illustrated in figure [2]. Even when the error tolerance of RANSAC was relaxed and sufficient number of iterations were performed, RANSAC identified lower number of inliers. From the figures [2] and [1] it can be inferred that by better classifying the inliers it is possible to decrease the error in estimates.

On the other hand, after a certain level of noise (beyond 1.4 pixels), the Sampson error of RANSAC started to saturate. This is due to the fact that the average error cannot go beyond the error threshold, hence decreasing the number of inliers. Whereas in our method, while still maintaining lower Sampson error, almost all of the points were identified as inliers. Also, this observation shows that for a higher level of noise increasing the number of inliers provides a better estimate.

Note that we had not intentionally introduced any outliers in this process. However, in the presence of noise correct matches with bigger drifts in location are essentially equivalent to outliers. Our method rejected only few such outliers. Initial estimation of RANSAC is solely driven by minimal number of points selected in each iteration. Such initial estimate not being accurate enough cannot incorporate other inliers. Furthermore, search in the linear space (unlike the exact rotation space in our case) is another reason for initial estimate deterioration. We believe that, since our method incorporates all the matches and searches in the exact rotation space, it is able produce more inliers while maintaining lower error in the estimate.

5.2. Performance in terms of running-time

Findings: The average time for execution of our method coupled with RANSAC is shown in table [1]. All the experiments were conducted in a system of 3.6GHz, 12 core (only one core running) with 64GB of memory. The table is further divided into times required for execution of parts of the algorithm. It was observed, higher the noise level is the slower the process becomes. The reason is that the initial estimate of inliers by RANSAC cannot classify all inliers mainly because of difficulty in distinguishing correspondences. As a result the pruning condition imposed by RANSAC is reduced in higher level of noise, resulting our method to perform an exhaustive search throughout the rotation space.

Table 1: Average running-time for RANSAC and for our method.

<table>
<thead>
<tr>
<th>Noise level (pixel)</th>
<th>RANSAC (secs)</th>
<th>SOS feasibility test only (secs)</th>
<th>Total time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.04</td>
<td>1.19</td>
<td>1.22</td>
</tr>
<tr>
<td>0.2</td>
<td>0.25</td>
<td>1.18</td>
<td>1.43</td>
</tr>
<tr>
<td>0.4</td>
<td>0.74</td>
<td>1.17</td>
<td>1.91</td>
</tr>
<tr>
<td>0.5</td>
<td>1.14</td>
<td>1.18</td>
<td>2.58</td>
</tr>
<tr>
<td>0.6</td>
<td>2.00</td>
<td>1.16</td>
<td>3.16</td>
</tr>
<tr>
<td>0.7</td>
<td>2.23</td>
<td>1.18</td>
<td>3.41</td>
</tr>
<tr>
<td>0.8</td>
<td>2.86</td>
<td>1.17</td>
<td>4.03</td>
</tr>
<tr>
<td>0.9</td>
<td>3.03</td>
<td>1.16</td>
<td>4.20</td>
</tr>
<tr>
<td>1.0</td>
<td>3.62</td>
<td>1.22</td>
<td>4.84</td>
</tr>
<tr>
<td>1.2</td>
<td>4.03</td>
<td>1.43</td>
<td>5.46</td>
</tr>
<tr>
<td>1.4</td>
<td>4.63</td>
<td>1.64</td>
<td>6.27</td>
</tr>
<tr>
<td>1.5</td>
<td>5.21</td>
<td>1.86</td>
<td>7.06</td>
</tr>
<tr>
<td>1.6</td>
<td>5.73</td>
<td>2.53</td>
<td>8.25</td>
</tr>
<tr>
<td>1.8</td>
<td>6.32</td>
<td>2.90</td>
<td>9.12</td>
</tr>
<tr>
<td>2.0</td>
<td>7.19</td>
<td>4.02</td>
<td>11.21</td>
</tr>
<tr>
<td>2.5</td>
<td>8.38</td>
<td>8.93</td>
<td>17.31</td>
</tr>
</tbody>
</table>

Figure 2: Number of inliers identified by the proposed and RANSAC methods from noisy image correspondences.
5.3. Performance in the presence of outliers

We extended the experiments of section 5.1 for dataset containing outliers. Basically, these outliers were generated by mismatching the point pairs. Multiple datasets for a constant noise of 0.9 pixels and fixed number of inliers 10, were generated. These datasets, after introducing different number of outliers, were tested with both RANSAC and our method. The distribution of number of inliers identified by these two methods are shown in figure 5.

Finding: It can be observed that our method identifies the correct number of inliers most of the times (74%) as compared to that of RANSAC (55%). On the other hand, RANSAC provides inconsistent number of inliers ranging from [7 to 12] with a wider spread. The mismatch being detected as inliers, i.e., 11 or 12 in our method, are mainly due to the allowed threshold on bound gap. We believe, this happens due to the randomly generated outliers. These outliers at times fall within the threshold of the estimated model experiments.

6. Conclusion

Although methods for computing geometric transformation between images are well known, such methods have been known to suffer from ambiguities due to presence local minima. Our method is based on Branching and Prune for global optimality. In this paper we put efforts to provide a mathematically guaranteed and globally optimal solution to geometric transformation estimation. The pruning mechanism in our method relies on the SOS feasibility. For this, the estimation problem is formulated into optimization over polynomial problem. By exploiting the polynomial’s degree and number of variables we opted for SOS optimization.

We showed the usefulness of our method for estimating rotation in purely-rotating camera. Using Cayley’s parameter for rotation parametrization, the polynomials induced by image correspondences resulted in a system of quadratic trivariate polynomials. We used our method to solve type of polynomials. Extensive experiments on synthesized data have demonstrated the validity of our proposed method.

References

Reconstruction of a 3D puzzle: application to the reassembly of a Latin writing from IV century

Florian Jampy, Eric Fauvet, Antony Hostein, Olivier Laligant and Frederic Truchetet
Laboratoire Le2i UMR 6306 CNRS, Université de Bourgogne,

Abstract

During the past ten year, the reconstruction of fragmented object becomes an important field of research in many areas, such as surgery, forensics, art restoration, material failure analysis. In archeology, reconstruction of broken artifact is a really time consuming task due to the omnipresence of fragmented like pots, murals, statues and inscriptions which leads to the elaboration of a convenient tool for the reassembling of theirs parts.

In this paper, we present an acquisition prototype for the 3D acquisition of an object contour based on an active stereovision system composed of a camera and a laser. The registration of each acquisition is done in parallel with a turntable allowing the rotation of the object. The propose solution have another camera for the 2D acquisition of the top face.

We propose a reconstruction method based on the experimentation of the complementarities of the fragments. We present a segmentation of the data in faces in order to process a large number of fragments.

1. Introduction

During the past ten year, the reconstruction of fragmented object becomes an important field of research in many areas, such as surgery, forensics, art restoration, material failure analysis. In archeology, reconstruction of broken artifact is a really time consuming task due to the omnipresence of fragmented like pots, murals, statues and inscriptions which leads to the elaboration of a convenient tool for the reassembling of theirs parts.

The object that must be reconstructed is a Roman tablet composed of an unknown number of slabs. Nowadays only 1200 fragments were found in Autun (France) most of them are 10 centimeter squared and a thickness of 8 to 20 millimeters. The tablet is mainly composed Pentelic marble. Each fragment has an arbitrary shape and in most cases a full or partial carved letter on the top faces. We can observe traces of burn on some fragments, cracks and holes are also present for the major part of the fragments. We can also observe some glue used for precedent reconstruction trial. The poor condition and the number of the fragments prevent all reconstruction attempts during the two last centuries.

The outcome of this work is to a solution which allows the digitization of the fragments this task will permit a virtual manipulation of the fragment. The Virtual manipulation is an important tool for the archeologists the mains reasons are the possibility to have all the fragments in the same place without monopolized a whole room for this task, it also avoid any time consuming manipulation and prevent to damage the fragments. The second outcome of the project is to provide a tool that assists the reassembling process by giving a measure of the congruency between two fragments.

2. Related work

2.1. Overview

In the literature, different approaches for reconstruction and automatic assembly methods can be found.

3. Reconstruction techniques

Most of the methods used for reassembly are based on the curve matching principle [1-3]. Assuming the object has 2D contours we are able to find experimentally a pair of corresponding contours by looking their outlines at sub-millimeter scale.

These approaches are generally time consuming and some multiscale methods [4-7] appear to palliate this issue.
A small variation of the method, Partial curve matching [8], more adapted to natural material and by the way more adapted to solve the problematic in archeology domain.

The reconstruction techniques are based on pairwise matching and then propagate the result most of them will fail if an entire fragment is missing or with badly damage fragments.

However an automatic reconstruction is required even if the result is partial. These results can be used by the archeologists and help to have an overview of the correct solution. This will provide a list of candidates for the manual validation.

4. Acquisition system

4.1. Overview

The acquisition systems must be transportable and usable by the Musée Rolin for later acquisition of fragmented objects and demonstration.

We decide to build our own prototype. In addition to fill the requirements, advantage is the possibility to have a flexible system which can evolve with the requirements of the project and palliate possible issue.

The actual acquisition system Figure 1 is composed of an active stereovision system with a camera (1) and a laser (2) this will provide the 3D information of the thickness contour. The turntable (3) provides the rotation necessary for the full digitization of the fragment. The camera (4) allows the 2D acquisition of the top.

4.2. Calibration

In our case the acquisition system and the rotation axis of the object are fixed according to the world referential, this configuration allows reducing the number of parameter to be estimated during the calibration part. In order to simplify the calibration step the laser planes include the rotation axis of the turntable.

The coordinate of the optic center are zero in the camera referential and given by \((x_c, y_c, z_c)\) in the world referential.

\[
\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 & t1 \\
0 & \cos \varphi & \sin \varphi & t2 \\
0 & -\sin \varphi & \cos \varphi & t3 \\
0 & 0 & 0 & 1
\end{bmatrix}\begin{bmatrix}
x_c \\
y_c \\
z_c
\end{bmatrix}
\]

\[
\begin{bmatrix}
t1 + x_c \\
t2 + y_c \cos \varphi + z_c \sin \varphi \\
t3 + z_c \cos \varphi - y_c \sin \varphi \\
1
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
0 \\
1
\end{bmatrix}
\]

For each single point in the image we can determine the related real coordinate. We choose the origin of the matrix on the CZ axis, \((u_0, v_0) = (0, 0)\)

\[
A[Rt] = \begin{bmatrix}
\alpha & 0 & 0 & 0 \\
\beta & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 & t1 \\
0 & \cos \varphi & \sin \varphi & t2 \\
0 & -\sin \varphi & \cos \varphi & t3 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
\alpha \\
\beta & 0 & 0 & 0 \\
0 & \sin \varphi & \cos \varphi & t3
\end{bmatrix} = \begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]

Figure 2: Axis position
We know that the point belong to the Laser plane so \( z = 0 \). We obtain the three following equations:

\[
\begin{align*}
    su &= ax + \alpha t1 \\
    sv &= \beta \cos \phi y + \beta t2 \\
    s &= -\sin \phi y + t3
\end{align*}
\]

We obtain

\[
\begin{align*}
x &= \frac{u(\beta t2 + t3 \beta \cot \phi)}{\beta \cot \phi + v} \\
y &= \frac{vt3 - \beta t2}{\beta \cos \phi + vsin \phi}
\end{align*}
\]

We have six independent parameters, as we don’t need the object position in the vertical axis. We can arbitrarily select the origin in the optical axis in order to remove \( t1 \) but we have to select the horizontal plane containing the optical axis. If the world contains horizontal line the one containing the optical axis are the only one to be horizontal in the image. That also implies the origin of the image referential must be select in the optical axis that means in the sensor center. The rotation axis \( y=0 \) is transpose in a vertical line of equation \( v = \frac{\beta t2}{\beta t3} \) in the image.

In order to calibrate the system we need at least 3 points, each of them gives 2 equations. We select the only horizontal line in the image that give.

\[ v_f = -\beta \cot \phi \]

For each vertical line in the image we have the relation between the world and image reference as.

\[ v = -v_f + y \]

\[ v = \frac{t2}{\cos \phi} + y \]

\[ v = -\frac{t2}{\cos \phi} + y \]

\[ v = \frac{t3}{\sin \phi} + \frac{t2}{\cos \phi} \]

\[ v = u\cos \phi \left( \frac{t3}{\sin \phi} + \frac{t2}{\cos \phi} \right) + v_f \]

The equation \( \frac{v-v_f}{u} = \cos \phi \left( \frac{t3}{\sin \phi} + \frac{t2}{\cos \phi} \right) \) represents the slope of the line. We have as many equations as we have lines in the image. We obtain another over determined system, which allow knowing \( cos \phi, \phi \) known we are able to compute \( \beta, t2, t3 \) and \( \alpha = \beta \) due to square pixels.

![Figure 4: Representation of the calibration grid in the image](image)

5. Reconstruction: 3D point cloud processing

5.1. Denoising

Before any later processing we need to remove the noise. Two classes of noise appear in the system.

The first is points who are not related to the object, the turntable below the fragment and the background above. The position of the points above the object is not known because of the variable thickness of the fragments. They are manually removed using a threshold adapted to the fragment thickness. An automatic removal is planned in the later stage of the acquisition prototype development.

The second class of point we need to remove is the random noise introduce by the system. It can be some isolated points inside or outside the fragment, it also possible to find a group of outlier probably due to some specular reflection. An automatic threshold is compute for each point according the density of is neighbor.

We obtain a denoised cloud of points ready to be processed.
5.2. Contours extraction

The main idea is to find two adjacent fragments by checking their mechanical compatibility. This solution is already used by the archeologist in order to find pair of fragments when texture information is not sufficient. Similar approach is curve matching for 2D object, in our case the thickness of the object can’t be neglect and with the 3D information of the contour we can transpose as a surface matching problem. The knowledge of the object will help the reconstruction by reducing the number of matching possibilities.

It is possible to find the matching between two fragments by using the full set of points from these two objects but it also computationally costly and inefficient. We know that, two adjacent fragments share at least one fractured face and in the majority of cases only one. So we can segment the cloud of points into faces in order to reduce the number of points used for each matching trial.

The first solution for the face extraction was to fit multiple planes in the object contour using Ransac algorithm [9]. The principle is finding the coordinate of the most significant plane in the point cloud, removing it and iterate. The results given using ransac were good approximation of the fragments face but we need to refine these results in order to remove aberrant plane and other artifact. In case of a single fragment the shape stay simple and the results were enough. Later we need to use more complex shape such as two or more merged fragments. Complex shape introduce new problem, these merged fragments can contain multiple faces in the same plane that create more instability with ransac and also start to be time consuming.

The second approach is an iterative algorithm minimizing simultaneously the distance at each points to the nearest plane. At each iteration, each point is affected in the nearest plane, if the distance from the nearest plane is above a threshold belonging to the object geometry; a new plane is created in order to regroup all the points without face affection. Faces are merged or split according the normal dissimilarity of the fitted plane, the proximity of points and the continuity of the faces. The main advantage is, no knowledge of the fragment is requiring.

An initialization is still needed with some information, instead of giving a random number of planes at random position it is better to give an initialization close to the solution we want. This will reduce the possibility to find any local minimum and the time to converge. For the initialization we use ransac but with less restrictive parameter.

Another advantage in comparison with the ransac based method is the possibility to keep previous results when we merged fragments. Due to that the initialization with ransac is done only once, for a new just digitized fragment. These possibilities give better result and drastically reduce the computational time.

5.3. Matching

As we have seen before using the only two faces of two different objects is similar to a registration approach, in this way we choose to use a slightly modified version of an Iterative Closest Points algorithm[10, 11]. The residual error of the ICP will give us a good knowledge about the matching of two opposite faces.

The residual value is generally hundreds time less for a good position than for any other possibility. The matching of two opposite faces is also less time consuming than using the whole fragments, for each matching trial it take around a second. According to this value we can estimate the time needed to compare a single fragment with the complete set of thousand others. Assuming each fragment has 5 faces, we need 25 trials for each pair of fragments that means nearly 7 hours are needed to compare a fragment with the whole set. For this reason we need to reduce the number of possibilities.
In order to reduce the number of trials for each fragment, we can use some knowledge we have about the tablet. Some of these clues are already used and implemented in the matching part.

We actually know the North/South orientation of each carved fragment, they are all scan in order to have the same orientation. This information allow in theory discarding all matching of faces with non opposite normal and remove the rotation along z axis. The matching is now a pure translation between the two fragments.

In practical we allow only few degrees of rotation in order to take account of any misplacement of the fragments during the acquisition process. In the same way we compute the matching of two faces even their normal are not strictly opposite. This important clue allow to reduce the number of matching for a pair of fragment to three most of the time and even less.

Another important information about the object is the thickness which is not the same along the tablet Figure 7. This difference of thickness is only few centimeters along the tablet but it staying noticeable on biggest fragments. We can assume two corresponding fragments must have the same thickness. This will significantly decrease the matching possibility in the whole set.

Figure 7: Global shape of the reconstructed object

6. Experiments

Below two digitized set of fragments for a total of 7 fragments. The results given Figure (8-9) are obtained by selecting automatically the first ranked result for each pairwise matching.

Figure 8: Two digitized set of fragments

7. Conclusion and future works

7.1. Acquisition system

We dispose of an acquisition prototype able to obtain the 3D information of the contour for each fragment and the 2D acquisition of the top face. 3D acquisition of the top face will be soon available. The acquisition time for a single fragment is approximately 3 minutes; this will take 60 hours in order to scan the full set of fragments. The planned development of the prototype should not increase this time.

7.2. Software

Regarding the software part, we successfully reconstruct the two set of fragments in an automatically way using only the 3D information of the contour. In the actual state this reconstruction takes few minutes. This time will increase with the data set but some implementations are planned in parallel to the acquisition system.
7.3. Evolution perspectives

The next step is to digitize a large number of fragments with the 2D/3D of the carved face and also the 3D acquisition of the contour in order to have a more representative sample of the tablet.

Some evolutions of the process are planned such as adding the character feature. We previously detail the improvement regarding an optical character recognition. It’s also possible to target some time consuming task of the process. Contour extraction is one of them, the sharp edge detection were not used here, discard due to a lot of smooth transition between the face but can give interesting result in combination with the actual iterative method. The matching part of the process is another option due to the number of use during the process. In [12] Huang, Q.-X., et al propose a modification off the ICP algorithm which can solve the problem of collision between two fragments, adapted to our problem it will probably give a better adjustments of correct matching and reduce the number of false positive by increasing the residual errors in this cases.

7.4. Generalization of the result

The given results could be applicable for some similar problems such as fresco, murals, tablets etc. The solution also could be adapted to any broken artifact with non smooth fractured faces. This approach is particularly well suit for artifact made of stone or similar materials. The reconstruction process can also be used for the case of gigantic object were the manual reconstruction become impossible due to the proportion of the fragments such as the inscription found at Pompeii composed of enormous plate of stone

References

Head Pose Estimation from Time-of-flight Data

Cansen Jiang

Abstract—This paper presents two methods to recover the Head Pose Estimation (HPE) from time-of-flight (ToF) data, namely Head Tracking and Discriminative Random Regression Forest (DRRF). The head tracking approach is achieved by combining 3D feature matching and tracking using 3D feature descriptors. The most relevant techniques to extract 3D key points and feature descriptor generators have been evaluated. The performance of the 3D feature descriptors is evaluated in terms of computation efficiency and robustness. The DRRF classifies the head patches, sampled from depth image, and regresses the head pose. The training and testing of DRRF is discussed.

I. INTRODUCTION

Head pose is an important indicator of a person’s focus of attention. For automotive industry and especially for Advanced Driver Assistance Systems, such information is of high importance in the Human-Machine Interface in order to increase car safety. HPE techniques have been profoundly studied in both 2D and 3D domains. Interested readers can refer to the survey presented in [3]. Recent 2D HPE techniques are very accurate and robust, achieving real-time performance under indoor environments with good lighting conditions [1][2]. However, under non controlled light conditions, e.g., outdoor environments, these approaches are less robust.

The fact that ToF cameras provide a depth map facilitates the head segmentation of the background and allows the computation of 3D features. The downside of ToF cameras is their relative low resolution and noise in the depth measurements. To estimate the HPE from ToF data several approaches were assessed. The most promising ones in point of view of requirements were implemented and evaluated in detail. The proposed 3D Feature Tracking approach extends the ideas from 2D feature tracking and matching. To track a head in 3D, a set of key points have to be detected from the face region. Then, 3D descriptors of these key points are used to find the correspondence between the template (or reference frame) and new frames. That is, a transformation matrix between the current frame and the template is estimated and decomposed into 3 rotations (Pitch, Yaw, and Roll) and 3 translations (X, Y, and Z).

HPE using DRRF is a combined classifier and regressor, which fulfills the head region classification and head pose estimation. The DRRF head estimator was trained using a large set of depth image annotated with head pose ground truth. The DRRF contains 7 trees, from where each tree has N leaves that vote for the head pose. Given a depth image, n patches are sampled and sent to the forest. These patches go through the trees from root to leaves. Then, head pose is estimated by taking the average of these votes.

The proposed head tracking requires a head segmentation and an initialization, where a frame of the forward looking driver is stored as a template. On the contrary, the DRRF method requires no initialization, but a long time training process.

The reminder of this paper is organized as follows: section II covers the most relevant HPE techniques from literature. Section III gives the scenario of our device setup and explains our head segmentation approach. Section IV states our head tracking algorithm, a brief introduction to 3D descriptors, and the performance of the head tracker. Section V discusses the DRRF approach in training and testing, and the performance evaluation. Concluding remarks and future work are given in section VI.

II. RELATED WORK

Various approaches had been explored and discussed to estimate the head pose orientation, as surveyed in [3]. In 2D computer vision, HPE mainly focuses on facial feature detection and localization. This information is used to estimate the relative changing of the head pose between video frames [5][6]. To improve the robustness of the tracking, i.e., robust to facial expression changing, [2] proposed to use an Active Appearance Model (AAM) that adapts the facial feature distribution to a flexible structure. Using AAM for tracking, the features points can be more precisely localized and tracked. More recently, [1][2] applied a Cylinder Head Model (CHM) to fit the head. The authors assume that the head can be modeled by a cylinder, which implies a very robust performance with wide angle HPE. The estimation of the CHM is constrained to an intensity consistent condition between two consecutive frames. The initialization only considers the Yaw rotation of the head. Furthermore, accuracy of the head position is highly related to the head size assumption.

On the other side and thanks to the recently progression of consumer-accessible depth cameras development, a noticeable interest has raised on 3D HPE [7][8]. Breitenstein et al. [7] proposed a 3D descriptor to localize the position of the nose tip. Assuming the nose tip is always seen, the geometry signature is calculated based on the local direction maximum. After extracting the shape signature, the head orientation can be estimated by comparing a set of annotated depth images to find the most similar target. Another simple but also computation heavy approach is HPE based on point cloud registration. [9] proposed a 3D head registration based on the

This work was supported by IEE S.A. Luxembourg.

C. Jiang is a MsCV student, University of Burgundy, 71200 Le Creusot, France cansen.jiang@etu.u-bourgogne.fr
A. Scenario of device setup

Point-to-Plane and Point-to-Projection ICP (Iterative Closest Point) algorithm. The performance of this method is slightly better than the HPE using geometry signature matching. On the other hand, Pashalis et al. [10] proposed an HPE algorithm based on the Particle Swarm Optimization. This method transforms the 3D points to a reference coordinate, and a new depth image is generated and superimposed with a reference depth image until it reaches the smallest error. The rotation and translation are decomposed from the transformation matrix. This approach achieves the best results so far in the literature. However, all these approaches are computational heavy and only can achieve real-time performance based on GPU (Graphic Processing Unit).

Differently, Fanelli et al. [8] constructed a head pose estimator based on the Random Forest framework with real-time performance. This method learns the human face appearance differences from a set of depth images, taking the advantage of large data off-line training. The classifier contains \( N \) random trees and \( n \) leaves, and mean value of these leaves’ votes gives precise prediction of the head translation and rotation. Moreover, this method is robust to partial occlusion and able to perform in real-time without using GPU.

III. HEAD SEGMENTATION

In order to track the head motion, segmentation is required. This section states the scenario of device setup and the segmentation method applied to robustly segment the head region.

A. Scenario of device setup

A pmdvision CamBoard nano camera produced by pmdtec. based on the ToF principle was used to acquire 3D data. The ToF camera has a resolution of \( 120 \times 165 \) pixels with \( 6mm \) lens and independent infrared light source with \( 30MHz \) modulation frequency. The frame rate of the ToF camera is \( 23.5fps \) with \( 2ms \) integration time. The ToF camera has robust performance in both indoor and outdoor environments. The camera was statically installed in a test car of IEE S.A., looking up (the angle between camera principle axis and the ground is \( 20^\circ \)) and facing the driver, as shown in Figure 1. The testing of our approaches was using the real data recorded inside the test car. However, since the ground truth acquisition device cannot be installed inside the test car, an experiment was conducted to acquire the ground truth data in the lab environment to evaluate the performance of the algorithm.

B. Local Minimum Segmentation

Though the camera is fixed, when driving, the seat and steering wheel can be adjusted by the drivers. In this case, part of the background cannot be removed by clipping. Also, the driver might be looking at different directions. In profile view, the normal face detection algorithms, such as Cascade Face Classifier, fail to detect the location of the face. In order to segment the head region, a local minimum based head segmentation approach was proposed.

3D point cloud from Time-of-Flight camera are sparse and noisy especially at the edges and complex geometric structures, such as the nose tip, eyes’ and mouth’s corners. To improve the quality the 3D point cloud, a set of preprocessing techniques, including background clipping, median filtering, head segmentation, jump edges elimination, and morphology processing, are applied in sequence on the depth image, see Figure 2.

As our camera is fixed, the background clipping is applied so that the objects locate farther than the background are removed. Therefore a clipping map has been generated by recording and temporal filtering of a depth video sequence of the car interior without a driver. The median filtering is also applied to remove the noise by taking a \( 3 \times 3 \) mask for each non-background pixel, with only foreground neighbouring pixels are evaluated.

The jump points, located in the sharp changed edges, are noisy and unstable. For a point \( p_i \) and its \( n \) nearest neighbors \( \{p_1, p_2, \ldots, p_n\} \), the average center-to-neighbor distance is calculated using \( av_i(d_i) = \frac{1}{n} \sum_{k=1}^{n}(|p_i - p_k|) \). For each 3D point, the average center-to-neighbor distance is used to model a Gaussian distribution using \( \mathcal{N}(\mu, \sigma) \), where \( \mu \) is the mean value and \( \sigma \) is the standard deviation. A pixel that has average center-to-neighbor distance larger \( d > \mu + 2\sigma \) is defined as a jump point.
In head segmentation, the head is assumed to be mostly near the center of the image. A local minimum, such as the nose tip, is the closest point to the camera within the head region. Starting from the image center, a sliding window (11 × 11 pixels) center is shifted to the position where has smallest depth value within the window. Until the smallest value locates in the center of the window, the local minimum is found, see Figure 4. And then, the head clustering is according to the head Anthropometry [12] which acclaims that 99% of the men head has vertical distance from the tip of the chin to the top of head are smaller than 213mm. In setting, neighbor distances smaller than 125mm are considered as head region. In next frame, the searching starts from the current local minimum position, assuming the head is not moving too fast. The head segmentation based on the local minimum works even far rotated head, and works in real-time.

The last step is to apply the morphology processing in order to remove the small objects in the image. As can be seen from Figure 3g, the connectivity of the objects is calculated and only the biggest object is kept, see Figure 3h.

IV. 3D HEAD TRACKING

After segmenting the head region, 3D feature based head motion tracking approach is applied, in order to estimate the transformation between the template and the new frames. This section introduces the head tracking approach and its performance evaluation.

A. Methodology

To track the motion of the head, the Intrinsic Shape Signature (ISS) key points on the head region are detected. Then the 3D feature descriptors on these key points are extracted and tracked by matching the similar feature descriptors on the face. In tracking the feature points, different algorithms are used to find the best tracking.

Firstly, a frontal face is taken as a reference frame and the ISS key points are detected from the template. The 3D descriptors of these key points are extracted and stored, and their positions are taken as the initial position of the tracking center in the next frame. These key points are tracked rather than all the points on the face. To find the corresponding points on the new frame, searching is conducted in a fixed neighborhood of (5 × 5) pixels centred with the previous position. Computing the descriptors only on the key points regions reduces the complexity of the calculation. The good matching point pairs are evaluated by the criteria \( D < d_{med} + 2\sigma \), where \( D \) is the distance between the point pairs, \( d_{med}, \sigma \) are the median point pair distance, and the standard deviation.

To match the key point descriptors, both the descriptor and Euclidean distances are considered. The point pair has both small descriptor and Euclidean distance are kept. To improve the robustness of the matching, the matching is based on a surface patch such that the point pairs are grouped and the point pairs that have similar properties in terms of descriptors distance and Euclidean distance are considered as correct matches. If a point pair’s properties are very different from other neighboring point pairs, the match will be discarded. In case of lost tracking, when the matches are less than 30%, restart the tracker from the initial state and find the matching points until enough matches are found, the tracking reestablished.

B. ISS key point detection

To improve the efficiency of the tracking, a set of key points, which are the landmarks like the nose tip, mouth corner, on the face are detected. ISS for key point detection was proposed by Zhong et al. [15]. The ISS is a local and view independent descriptor for fast pose registration that takes an intrinsic reference frame. The intrinsic reference frame is calculated by taking the Principle Component Analysis (PCA) of a weighted neighbor point scattering function. The weights are applied to give more contributions of the points located in sparse region, in order to compensate the uneven sampling of the point clouds. Figure 5 shows the detected key points using ISS covering most of the landmarks on the face, spreading most of the area on the face. However, the 3D Harr detector is more likely to detect the key points on the boundary as the Harr detector detects the sharp changed features, such as corners.

C. 3D Feature Descriptor

Unique Shape Context descriptor (USCD) [16] is an extension of the 3D Shape Context Descriptor [17] (3D SCD)
by adding the directional constraints on the 3D SCD with a local reference frame. The computation of USCD consists of two parts: the calculation of local reference frame and the extraction of shape context descriptor. Taking the center point as the origin of a spherical coordinate, the 360° angles are divided into \( n \) bins with \( k \) sections. A histogram is built by counting the points that fall in the region. The uniqueness of the descriptor results from the local reference frame.

Similarly, the Unique Signatures of Histograms of Orientation (SHOT) \([18]\) feature descriptor is a local histogram that describes the distribution of the point clouds within a given radius of sphere. A support sphere is built based on a local reference frame and divided into \( N \) different volume sections. The local reference frame calculation is similar to the USC descriptor. A 32 bins histogram is the accumulation of 3D points locate within the specific region.

The Fast Point Feature Histogram (FPFH) \([13]\) feature descriptor involves from the Persistent Point Feature Hitograms \([19]\), becoming more robust and faster. The FPFH generalizes both the surface normals and the curvature estimates so that it is very robust in detecting the curvature changing regions. For each point, the normal of the point is estimated using PCA. Then a Darboux frames is applied to construct a feature histogram that describes relationship between a center point and its neighbors.

### D. Evaluation

An OptiTrack Motion Capture System, see Figure 1, was used to acquire a set of ground truth data. The OptiTrack System tracks the movement of the optical sensors (marker) that are attached on top of the head. The motion of marker is captured and transformed into head translation and rotation. During the data acquisition, 6 candidates participated to the experiment. They were 5 men and 1 woman, with half of them wearing glasses. Each of the participant moved his/her head freely in all directions and small translations. The rotations in Yaw, Roll and Pitch range from \( \pm 95°, \pm 75°, \pm 75° \) respectively. The total number of the video sequence last for about 6,000 valid frames.

The performance of the head pose tracking is compared to the ground truth data. The ground truth data covered the wide range of movement regarding to the Yaw, Pitch and Roll rotation angles. To evaluate the performance of the tracking using different descriptors, only one direction of rotation was validated at a time.

Figure 7 shows the head rotation around Yaw axis ranging from \( \pm 95° \) with Pitch and Roll angels slightly changed. For Yaw rotation, the tracking failed from frame 20 to 150. But tracking recovered from frame 150 to 210 and 300 to 370, where the head rotations referring to the template were small within the range of \( +10° \) to \( -30° \). The tracking was failing because the rotation of the head was too big, so fewer corresponding features could be seen in the new frame. Regarding to Pitch rotation, the tracking was not correct. Since the ToF camera was set looking up to the head about \( 20° \) with the ground, small rotations in the Pitch caused big difference for the ToF data. Also, as the ToF camera is very sensitive to depth changing, the Pitch rotation increases the depth values of the lower part of head while decreases the higher part of the head. In this case, the sampling of the 3D points on different parts of the head is not even and very different from the template. For the Roll rotation, the tracking was not working because the changing of the rotation is too fast and the tracking got lost at the very beginning.

The performance of USC, FPFH, and SHOT descriptors are evaluated using the same data set. The USC descriptor performances slightly better than the FPFH and the SHOT descriptor in terms of robustness and efficiency. The test was run in a i5-3210M, 2.5GHz and 8.0GB machine, and the USC, SHOT and FPFH can reach a frame rate about 3.5\( \text{fps} \), 2.0\( \text{fps} \), and 2.0\( \text{fps} \) by calculating 300 feature descriptors. The performance of the tracking can be improved by optimizing the code and reducing the searching space of feature descriptors.

### V. DISCRIMINATIVE RANDOM REGRESSION FOREST

DRRF head pose estimator is a machine learning based approach that predicts the head pose. The DRRF approach is discussed in this section, and the performance was examined by using the ETH data base \([8]\).

#### A. Methodology

The random forest is an ensemble of decision tree classifiers, which fulfils the head classification and the pose estimation. In head classification, the leaves of the trees contain the probability \( p(\text{class} = \text{head}|P) \) of being a head leaf. The
head leaves give vote for the head pose in regression. In tree growing, a large data set in both positive and negative cases are randomly sampled. A positive sample is defined as having more than 30% of the area belonging to the head. Starting from the tree root, the testing set \{U\} is split to left subset and right subset based on a feature response function. At each node, a binary pool of splitting parameters \(T_{(i,F_1,F_2,\tau)}\) are generated randomly. The testing set is split by the whole binary pool and the optimal split is the one with minimum class uncertainty.

Once the subset \(\{S\}\) reaches a leaf node, which has maximum depth or less than 20 test samples, the probability \(p(\text{class} = \text{head}|P)\) of the leaves belonging to foreground is estimated and stored. Only the foreground leaves are used for regression voting. Moreover, the mean values and trace of the covariance matrix of head pose parameters \(\theta_{x}, \theta_{y}, \theta_{z}, \theta_{xy}, \theta_{xz}, \theta_{yz}\) are also stored. The 6-vector contains the translation (in mm) from the patch center point to the head center position, and the rotation angles (in degree) in Yaw, Roll and Pitch of the head. The testing set reaching the leaf are modelled as a multivariate Gaussian distribution. The higher value of trace represents the higher uncertainty of prediction.

Regarding to pose estimation, a regression process is conducted based on calculating the mean values of the votes from all leaves. Running the classifier, the target patches are sent to the classifier, going through the whole tree until they reach the leaves. Thanks to the classification, the leaves contain the probability \(p(\text{class} = \text{head}|P)\) of the patch belonging to the head. The multivariate Gaussian model predicts the offset from patch center to the head center, and the head rotation as well. Taking the mean value of the predicted head center and head orientation, the head pose is estimated.

### B. Training

Given a set of depth images labelled with head center and orientation, \(k\) trees are grown by randomly select \(M\) training images and \(N\) negative and positive patches \(\{P_i = (I_i, c_i, \theta_i)\}\) from each image. The patches are labelled with the class property \(c_i\), and the 6-vector. Starting from the root, the training set will be split into left and right child nodes based on the feature responds function [8]:

\[
|F_1|^{-1} \left( \sum_{q \in F_1} I(q) \right) - |F_2|^{-1} \left( \sum_{q \in F_2} I(q) \right) > \tau
\]

(1)

where \(F_1, F_2\) are two rectangle asymmetric regions subsampled from the training patch, and \(|F_1|, |F_2|\) represent the size of the rectangles, \(\tau\) is the threshold. \(I(q)\) is the feature channel randomly selected from the feature vector (depth image). From Figure 8, two feature vectors are randomly generated from the depth patch, the size of the feature vectors are constrained to maximum 50% of the patch.

To evaluate the split, the classification and regression can be measured separately. In classification, the uncertainty of the class label should be minimized, which is described by the measure function [8]:

\[
U_c(P|t^k) = |P_L| \sum_{q \in F_L} p(c|P_L) \log(p(c|P_L) + |P_R| \sum_{q \in F_R} p(c|P_R) \log(p(c|P_R))
\]

(2)

where \(|P_L|, |P_R|\) is the number of the left and right subset, \(p(c|P)\) is the ratio of the patches belonging to class \(c_i \in \{0, 1\}\). Equation 2 measures the performance of node split in discriminating the foreground patches and the background patches.

In regression, the measure function aims to maximize the information gain to favor the regression accuracy. The measure function is defined as [8]

\[
U_R(P|t^k) = \log(|\Sigma^c_r| + |\Sigma^o_r|) - \sum_{i \in \{L,R\}} w_i \log(|\Sigma^c_r| + |\Sigma^o_r|)
\]

(3)

where \(U_R(P)\) is the differential entropy of set \(P\) and \(w_L, w_R\) is the ratio of the left and right subset of the split. \(\Sigma_r\) is the translation covariance matrix of the 6-vector, and \(\Sigma\) is the determinant of the covariance matrix. The covariance between the translation vectors and the head rotation angles is assumed to be block-diagonal. By minimizing the determinant, one can maximize equation 3 to minimizing the uncertainty of the regression. The equation 2 and 3 can be combined into an weighted optimization function [8]:

\[
\arg\max_k \left( U_c + (1.0 - e^{-d/\lambda}) U_R \right)
\]

(4)

where \(d\) is the depth of the node and \(\lambda\) is the scale factor that controls the weight of the regression. Taking the exponential, the deeper the tree grows, the higher weight of the regression function.

### C. Evaluation

The classifier was trained using 80% of the data set and 20% was used to test the classifier without cross-validation. The training was run in an i5-3210M, 2.5GHz and 8.0GB machine, and it took around 10 hours to build a 7 trees classifier. The testing results in terms of head translation and rotation are compared in Figure 9. The average absolute errors in X, Y, Z axis translations are 58mm, 51mm, 24mm respectively, where X axis is horizontal and Y axis is pointing down and vertical to the ground, Z axis is pointing out the camera. The error of Z axis is much better even though the variation of the Z values are bigger. For prediction in X and Y direction, the mean distance is less than 60mm, which
means that, in most cases, the head region can be correctly determined.

Regarding to the regression of the rotation, the average absolute errors in Pitch, Yaw and Roll are 30.7°, 32.6° and 9.6° respectively. Considering that the average absolute error within 6° is correct estimation, the regression can reach 5.2% correct rate. Some estimation result using the classifier is shown in Figure 10.

![Fig. 10. DRRF in HPE result](image)

VI. CONCLUSIONS AND FUTURE WORKS

A dedicated robust and fast head segmentation method for 3D ToF camera has been developed. The Local Minimum value based segmentation works robustly even with far rotated head in real-time. For determining head orientation and translation, the feature base head tracking approach was implemented and tested, different feature descriptors were applied to evaluate their performance, in terms of accuracy and time consuming, in head tracking. Experiment demonstrated the template based tracking is robust but limited at small rotation. Discriminative Random Regression Forest (DRRF) head pose estimator was implemented to fulfill the tasks in head classification and head pose estimation. Result showed that the head region can be correctly classified, while the regression in head center and orientation need to be improved.

Future work to overcome the limitation of head tracking by introducing more templates. For instance, at every 30°, templates are stored and registered to a zero rotation template. When tracking results show that the head rotates more than 30°, the tracking template with be changed and new frames will be tracked based on the new template. For DRRF head estimator, the units of the head translation and rotation should be normalized, and the calculation of the covariance matrix determinant of the optimization function should be explored to avoid the singularity problem.

VII. ACKNOWLEDGMENTS

The authors gratefully appreciate the support and useful comments from Dr. Bruno Mirbach and Dr. Frederic Garcia, and all the colleagues at IEE S.A.

REFERENCES

Quasi-Isometric Volume-Based Shape-from-Template

Shaifali Parashar, Daniel Pizarro and Adrien Bartoli

Abstract—Reconstruction of 3D objects from images is a key problem in Computer Vision with important applications. Reconstruction for rigid objects is mainly solved with Structure-from-Motion (SfM) techniques. However, rigid-based methods fail when applied to objects undergoing deformations, such as the human body or a piece of cloth. Reconstruction problems such as Non-Rigid Structure-from-Motion (NRSfM) and Shape-from-Template (SfT) have been recently studied for particular deformations and thin-shell materials or surfaces. This thesis studies SfT for deforming volumes. In SfT, a template is known and the objective is to find the deformation from a single input image. It is a challenging problem, as, for opaque objects, only a portion of the objects surface is visible. We model deformations with a Quasi-Isometric model that softly imposes local rigidity, allowing deformations. Quasi-Isometric SfT for volumes has not been studied before. This thesis presents three main contributions: First, we describe the problem with a system of partial differential equations and differential inequalities. We show that the system does not admit point-wise solutions. Second, we propose to find a global solution transforming the system into a variational optimization problem that can be solved using unconstrained iterative optimization. Third, we present a method to find an initial solution based on local rigidity propagation and surface isometric reconstruction. Experiments with real and synthetic data show that our method is very accurate in recovering large volume deformations. They also verify that our initialization method is always very close to the correct minimum.

I. INTRODUCTION

Recovering 3D from images is an important problem in computer vision that has been successfully solved in case of rigid environments with Structure-from-Motion (SfM). However, rigid-based methods fail when objects undergo deformations, such as piece of paper or the human body. Non-rigid reconstruction methods represent an important challenge with a wide spectrum of applications (e.g., medical imaging, entertainment industry, etc.).

We highlight two important non-rigid reconstruction problems: i) Non-Rigid Structure-from-Motion (NRSfM) [5] [3] [7], where given a set of images of a deforming object, we obtain the set of shapes and ii) Shape-from-Template (SfT) [16] [1], where given a single image and a template of the object, the objective is to obtain the shape. Our work falls into latter category, extending the shape estimation to volumes.

Most of SfT methods apply only to thin-shell materials by modelling surfaces deformations. In SfT deformations has to constrained, otherwise the problem is ambiguous. We distinguish two different approaches to model surface deformations: i) learning-based methods [6] [17], where the space of possible shapes or deformation is learn from data and is assumed to live in a low dimensional space and ii) physics-based models such as isometric [16] [1], conformal [1] or linear elastic deformations [11]. Isometric deformations have received most of the attention as they are good models for a wide range of thin-shell materials. Recently [1] proved that SfT for isometric deforming surfaces is well-posed and can be solved analytically.

Volume-based SfT is a hard problem as only a partial view of the volume’s external surface is projected in the image. In this paper, we study the SfT problem for volumes that deform quasi-isometrically. Isometric deformations between volumes are rigid transforms. Quasi-isometry relaxes rigidity to allow deformations, while preserving, up to certain degree, the structure of the object. This model has been widely used in graphics to perform mesh editing of animated characters [19].

Contrary to surfaces, volume-based SfT is almost completely unexplored. To our knowledge, only a recent paper [18] has proposed volume reconstruction from a single image. However, their method only works for shapes that can be inferred from its silhouette, which poses important constraints on the topology of the object and the deformations. In addition, their reconstruction is based on an orthographic projection which cannot be applied in all imaging conditions.

Our contributions are three-fold: First, we show that quasi-isometric volume based SfT for perspective projections can be described as a system of partial differential equations involving differential inequalities. We show that in general the system is not locally solvable. Second, we introduce a criterion that allows to convert volume SfT into a constrained variational optimization problem, where we impose the surface to be as isometric as possible. We show how to discretize and to relax the constraints to convert the variational cost into an unconstrained non-linear least squares optimization problem. Third, we present a method for initializing the solution based on local isometry propagation on the volume and surface isometric SfT. The initialization method involves linear least-squares only. Experiments with real and synthetic data show that our method is very accurate capturing volume deformation of objects. They also verify that our initialization method is always very close to the correct minimum.
II. Modelling Volume SfT

Fig. 1 shows a general diagram of the volume reconstruction problem that involves: i) the 3D template volume $V_T \in \mathbb{R}^3$, ii) the unknown deformed volume $V_S \in \mathbb{R}^3$, iii) the image plane $\mathcal{I}$ projecting the visible surface $\mathcal{S}$ of $V_S$ and iv) a 2D flattening $\mathcal{F}$ of the external surface $\mathcal{S}$ of the template $V_T$. Deformation between $V_T$ and $V_S$ is given by the unknown mapping $\psi \in C^2(\mathcal{F}, \mathcal{S})$. We assume opaque objects and thus image $\mathcal{I}$ projects a surface $\mathcal{S}$ that belongs to the exterior of volume $V_S$. The rest of the volume is self-occluded. Note that self-occlusions depend on $V_T$ and can be estimated during registration [8] [14]. We define as $\mathcal{F}$ the corresponding visible surface in the template. We use $\Pi$ to denote perspective projection in image coordinates normalized with respect to the camera’s intrinsic parameters:

$$\text{Given } Q \in \mathbb{R}^3 \text{ then } \Pi(Q) = \left( \frac{Q_x}{Q_z}, \frac{Q_y}{Q_z} \right)^\top. \quad (1)$$

![Fig. 1. General diagram of Volume SfT](image)

Note that $\Pi$ involves geometric projection but does not deal with self-occlusions directly. We define $\Delta \in C^2(\mathcal{F}, \mathcal{S})$ as the embedding that describes surface $\mathcal{S}$ from a 2D flattening $\mathcal{F} \in \mathbb{R}^2$. $\Delta$ and $\mathcal{F}$ are known. From $\mathcal{F}$, we parametrize the unknown surface $\mathcal{S}$ using the embedding $\phi \in C^2(\mathcal{F}, \mathcal{S})$. Following the consistency between mappings we have $\phi = \psi \circ \Delta$.

Finally we define as $\eta \in C^2(\mathcal{F}, \mathcal{S})$ to the known registration warp between $\mathcal{F}$ and the image. The warp $\eta$ establishes the reprojection constraint on $\psi$.

$$\eta = \Pi \circ \phi = \Pi \circ \psi \circ \Delta. \quad (2)$$

A. Isometric Surface Deformations

Isometric deformations have proven to be a good model for real thin-shell materials such as paper or a piece of cloth [16] [1]. To illustrate them we follow [1]. Let us assume that the surface $\mathcal{F}$ deforms isometrically, resulting in $\mathcal{S}$. The isometric mapping between $\mathcal{F}$ and $\mathcal{S}$ preserves surface first fundamental form [10]:

$$J_\psi J_\psi = J_\Delta J_\Delta, \quad (3)$$

with $J$ an operator giving the first-order partial derivatives. By taking first derivatives of $\phi = \psi \circ \Delta$ we get $J_\phi = (J_\psi \circ \Delta)J_\Delta$. Introducing this result in Eq. (4):

$$J_\Delta (J_\psi \circ \Delta)^\top (J_\psi \circ \Delta)J_\Delta = J_\Delta J_\Delta \Rightarrow (J_\psi \circ \Delta)^\top (J_\psi \circ \Delta) = I_{3 \times 3}, \quad (4)$$

where we derive that $(J_\psi \circ \Delta)$ is an orthogonal matrix. Note that $J_\psi \circ \Delta$ only operates in the surface $\mathcal{F}$ which allows the surface to bend while preserving distances on the surface. This distinction is important when defining volume isometry.

I) SfT for Isometric Surfaces is Locally Solvable: In surface-based Shape-from-Template [1] the reprojection constraint and the surface isometric deformation constraint of Eq. (4) are combined to obtain the unknown embedding $\phi$:

$$\text{Find } \phi \text{ s.t. } \begin{aligned} J_\psi J_\phi &= J_\Delta J_\Delta \quad \text{Deformation Constraint} \\ \eta &= \Pi \circ \phi \quad \text{Reprojection Constraint} \end{aligned} \quad (5)$$

Eq. (5) is a system of Partial Differential Equations (PDE). Recently [1] shows that SfT system (5) can be solved locally and analytically, which means that we can obtain point-wise solutions without solving a boundary value problem.

B. Quasi-Isometric Volume Deformations

If $\psi$ is an isometric transformation between volumes $V_T$ and $V_S$ we have:

$$J_\psi J_\psi = I_{3 \times 3}. \quad (6)$$

Condition of Eq. (6) in volumes makes $\psi$ a rigid transformation (see Mazur-Ulam theorem [15]). Given a point $Q \in V_T$, we thus have $\psi(Q) = RQ + t$, where $R$ is a constant orthogonal matrix and $t$ is a constant translation vector.

In quasi-isometry [9] we relax the condition Eq. (6) to obtain the following deformation constraint:

$$\left\| J_\psi J_\psi - I_{3 \times 3} \right\|_p^2 \leq c \quad c \geq 0, \quad (7)$$

where $p$ is any matrix norm (e.g., $p = F$ for Frobenius norm) and $c$ is a given constant that controls how rigid the mapping is. We refer the reader to [9] for a formal definition of quasi-isometric mappings.

III. Volume Based Quasi-Isometric SfT

We present our volume based SfT approach for quasi-isometric mappings, showing that it involves boundaries and partial differential inequalities. The objective in volume-based SfT is to find the deformation map $\psi$ that transforms volume $V_T$ into the unknown volume $V_S$ given that we partially observe it in $\mathcal{I}$. We combine quasi-isometric deformation constraint in Eq. (7) and the reprojection constraint of Eq. (2) to get:
Find \( \psi \) s.t. \[
\begin{align*}
\left\| J_{\psi}J_{\psi} - I_{3 \times 3} \right\|_F^2 & \leq c \\
\eta &= \Pi \circ \psi \circ \Delta
\end{align*}
\]

Deformation Constraint

Reprojection Constraint

Eq. (8) is a system involving Partial Differential Inequalities [13]. Finding local solutions of Eq. (8) is not possible for two reasons: i) the system is a boundary value problem as the reprojection constraint only involves a surface that is a subset of the volume \( S \in Y_S \) and ii) the deformation constraint is an inequality which means that we may have infinite solutions.

A. A Variational Solution for Quasi-Isometric SfT

As was mentioned before, we cannot find point-wise solutions of Eq. (8). We propose to convert Eq. (8) into a variational optimization problem where we obtain a global solution for \( \varphi \). We add the following criterion: we select a function \( \varphi \) that is as isometric as possible, where Eq. (7) is minimized for all points in the volume. This condition yields the following optimization problem:

\[
\psi = \arg\min_{\psi} \int_{Y_T} \left\| J_{\psi}J_{\psi} - I_{3 \times 3} \right\|_F^2 \quad \text{s.t.} \quad \eta = \Pi \circ \psi \circ \Delta
\]

The analytical solution for Eq. (9) is in general not achievable as it involves integrals, non-convex functionals and equality constraints. However, we can transform Eq. (9) into an unconstrained non-linear least squares program that we can solve numerically.

We first define the deformation functional \( \epsilon_d[\psi] \) that approximates the As Isometric as Possible cost by discretizing \( Y_T \) and substituting the integral by a sum:

\[
\int_{Y_T} \left\| J_{\psi}J_{\psi} - I_{3 \times 3} \right\|_F^2 dY_T \approx \sum_{p \in Y_T} \left\| J_{\psi}(p)J_{\psi}(p) - I_{3 \times 3} \right\|_F^2 = \epsilon_d[\psi]
\]

Instead of imposing the reprojection constraint exactly, we use a lagrangian relaxation that employs the following functional over a discretization of \( Y_T \):

\[
\sum_{p \in \mathcal{P}} \left\| \eta(p) - \Pi(\psi(A(p))) \right\|_2^2 = \epsilon_R[\psi]
\]

1) Optimizing the Cost: To find a solution to Eq. (12), we propose a parametric representation of the solution \( \psi \in C^2(Y_T \times \mathbb{R}^n, S) \), where \( n \) is the dimension of the parameter space. Let \( I \in \mathbb{R}^n \) be the parameter vector and \( Q \in Y_T \) we have \( \psi(Q, I) \in Y_S \). We have multiple choices for representing \( \psi \) such as the popular linear basis expansion warps (e.g., NURBS [12], TPS [2], B-Spline [4], Tetrahedron Mesh displacements, etc.).

The cost (12) is then optimized in function of the parameters \( I \) using an iterative optimization method, such as Levenberg-Marquardt. Iterative methods can be arbitrary accurate but as Eq. (12) is non-convex the iterative algorithm is prone to be stuck in a local minimum.

2) Initialization with Greedy Propagation of Local Isometry: We propose an algorithm that provides an accurate initialization \( \psi_0 \) to the non-linear refinement algorithm. It is based on surface SfT followed by a greedy propagation of local quasi-isometry. It is composed of the following steps:

Step 1) Discretizing the domains: First, we discretize the template volume \( Y_T \) with a set of \( N \) 3D points that we call \( \mathcal{P}_T = \{P_1, \cdots, P_N\} \). We denote as \( \mathcal{P}_V \subset \mathcal{P} \) the subset of \( N_S < N \) points corresponding to surface \( \mathcal{S} \). From \( \mathcal{P}_V \) and knowing the flattening function \( \Delta \), we compute the corresponding set of 2D points \( \mathcal{P}_F = \{P_1, \cdots, P_{N_S}\} \) in the flattened domain. Using a triangulation method (e.g., Delaunay) we define a tetrahedron mesh with \( \mathcal{P}_F \) as vertexes.

Step 2) Surface SfT: We assume for the initialization that the deformation between \( \mathcal{S} \) and \( \mathcal{S} \) is a surface isometry. We can thus solve system (5) using (1) to obtain the set \( \mathcal{P}_F \) of points belonging to surface \( \mathcal{S} \).

Step 3) Mesh Propagation: For a given tetrahedron, we denote as \( P_1, P_2, P_3, P_4 \) the four vertexes in \( Y_T \). Let us assume that three of the vertexes \( (n_1, n_2, n_3) \) belong to the surface \( \mathcal{S} \). From the previous step, we know their corresponding points in \( \mathcal{F} \), namely \( Q_{n_1}, Q_{n_2}, Q_{n_3} \). We estimate a rigid transform between the two sets that is used to obtain \( Q_{n_4} \). As a sanity check, we test that the mean distance between the obtained points is similar to that in the template, otherwise the point is discarded. Once we’ve finished with all tetrahedrons belonging to \( \mathcal{F} \), we repeat the process by recursively finding all tetrahedrons from which we have three known corresponding vertexes in \( \mathcal{S} \). We repeat the process until the last tetrahedron with a missing vertex is found. The result of this step is the deformed set \( \mathcal{D}_F \subset Y_S \).

Step 4) Obtaining \( \psi_0 \): From the two sets \( \mathcal{P}_V \) and \( \mathcal{D}_F \) we obtain \( \psi \) by solving the following optimization problem:

\[
\psi_0 = \arg\min_{\psi} \{\alpha \epsilon_s[\psi] + (1 - \alpha) \epsilon_{smth}[\psi]\} \quad 0 < \alpha < 1,
\]

where \( \epsilon_s[\psi] \) minimizes the transport error between the two sets and \( \epsilon_{smth} \) is a smoothing term (e.g., Bending Energy). Note that system (13) involves linear least-squares.

IV. EXPERIMENTS AND RESULTS

In this section, we will describe the experiments that we performed to test our method and discuss the results obtained. We conducted the experiments in order to test the
limitations of our method. We implemented our method in MATLAB. We conducted experiments with synthetic data and two sets of real data with different geometries: a book and a tube.

a) Experiment with synthetic data: We consider a cuboid of dimension $20 \times 20 \times 2.5$ cm and deform it at an angle of 20 degrees. We use 100 points on the surface for the experiment. The results are shown in Figure 2. We measured the 3D residual error in mm as the average distance between the simulated and the reconstructed 3D points. For each configuration, we kept the average of the 3D residual error over 20 trials. Our initialization method provides a good starting point for the optimization setup, the optimal solution is achieved within 3 or 4 iterations only. We tested the behaviour of our method in different conditions to illustrate the effect of change of number of keypoints, deformation angles and the noise. The graphs are shown. We see that the accuracy of our method decreases with increasing noise and deformation angle and increases with more number of points used in SfT.

b) Experiment with real datasets: For the experiments on real datasets, we use a set of 20-30 images with different poses of the object to create the 3D models of the objects. These models can be built by laser scan, structured light or any other 3D reconstruction technique. We have built these models with the help of a commercial Structure-from-Motion software, which gives a very accurate model for the objects.

Experiment with the book We conducted two experiments on this dataset. The results are described in Figure 3. The views for the template and the object with 208 and 327 feature points are shown. For both experiments, we use these two images to obtain the ground truth for reprojection (the visible surface of the template) and generalization (the non-visible surface of the template) surfaces. In the figure, we see that the results of Experiment 2 are slightly better than Experiment 1 because of the number of feature points. The error at the generalization surface is larger due to the accumulation of error in propagation step.

Experiment with the tube We conducted three experiments on this dataset. The results are described in Figure 4. The views for the template and the object with 108, 139 and 126 feature points are shown. For Experiment 1 and 2, we use the two images to obtain the ground truth for reprojection and generalization surfaces. In the figure, we see that the results of Experiment 2 are slightly better than Experiment 1 because of the number of feature points. The error at the generalization surface is larger due to the accumulation of error in propagation step. For Experiment 3, we increased the deformation in the object in order to test our method in extreme conditions. We see that the errors at reprojection and generalization have almost doubled. This is in accordance with the deformation of the object.

V. CONCLUSIONS

We presented a method to construct the complete volume of a 3D object using a single image and an object template.
We extend the surface isometry to volumes considering the quasi-isometric deformations. We use SfT to initialize the points on the surface, propagate the deformations using tetrahedral mesh and refine the results by using data and deformation constraints. Our experiments show that our method achieves a good accuracy in recovering the visible and non-visible parts of an object. We also show that our method handles large deformations very well. Our initialization method is very close to the optimal minimum.

VI. ACKNOWLEDGMENTS

I would like to express my gratitude to my supervisors: Daniel Pizarro and Adrien Bartoli for their guidance and support. I am very thankful to Toby Collins for his constructive feedbacks at many stages. I would also like to thank Ajad Chhatkuli, Luis Tobais and Rahat Khan for their help.

Special thanks to the MSCV/VIBOT professors and friends for being a part of this Masters, it has been a journey of knowledgeable and wonderful two years.

REFERENCES


Abstract—In this paper, we propose a graph-based representation of three features devoted to category object recognition, namely skeleton, superpixel and self-similarity of an object. These features are computational models interpreting gestalt laws in terms of shape of an object and appearance of an object through colour distribution. The skeleton defines the structure of the object, superpixels provide the global appearance of the object and self-similarity provides the internal geometrical information of parts with respect to others in the object. We represented these features as graphs, namely skeleton graph, superpixel graph and self-similarity graph and we performed graph matching pairwise and then we propose a new method to represent all these graphs in one graph inspired by hypergraph. The results obtained over Caltech 101 database show good and promising results. Graph Theory applied in object recognition is a challenging task and this work proposes new ways of addressing such problem under the point of view of computer vision domain.

1. INTRODUCTION

One of the most interesting and amazing things in the human life is vision. The ability to see around us and understand the world is an interesting fact. Most of us don’t even think of how it works. How can we see and understand the things around us. Our visual system is quite interesting such that we sense the world and get the very minute detailed information about the environment effortlessly. Vision is a complex system or process that has many components for interaction, which are involved. For example, our vision recognizes the object by or with color analysis, texture of the object, shape and depth and we use visual information for locomotion, recognition and manipulation [1].

We can identify the things, recognize people and classify things most of the time easily. All these things are most of the time effortless for human beings and actually we should say for most of the living beings such as animals. The tasks, which are trivial for most of the living beings even when the conditions are not favorable, are the basic problems in computer vision where still struggle is going-on to achieve solutions. One explanation for this resides in the so-called semantics gap between low-level content and higher-level concepts [2].

Generally, machines are able to extract and process low-level visual information but cannot interpret the same data for a given situation. While humans are doing this process with ease and even we don’t realize that. This is because of the visual system in the living beings, which has been evolved over many centuries. There are numerous attempts going-on in this process to teach the machines to understand the scene with different type of features. To some extent we are even trying to replicate the human visual system yet there is no success to a notable extent. So still the question remains the same: How can we build a system that sees and perceives the world as we do?

The work done and time taken by the visual system (eyes and brain) of humans in this context is negligible while machines perform an exceeding work in processing the information. In this process, there are many experiments conducted and still in process about the living beings visual system in areas such as neurophysiology, artificial intelligence and computer vision. Along the years the three psychological theories of visual perception, which aimed to explain how things look as they do to us, were Structuralism, Gestaltism and Ecological Optics [3].

Among these theories, the Gestalt movement led by Max Wertheimer, Wolfgang Kohler and Kurt Koffka in 1935 gained a lot of respect from scientific community due to the systematic description of its approach and beliefs which included many examples based on real-world experiments [3]. According to Gestaltists the whole is different from the sum of its parts, which could be interpreted, as that humans perceive the whole structure first rather than the details. The innovation in Gestalt theory was to describe this phenomenon through a series of mechanisms for perceptual organization known as Gestalt Laws. Therefore, visual perception in the Gestalt point of view obeys to some kind of organization from which both; the whole and its parts have importance for understanding the scene.

There have been different types of structures used for object recognition such as histogram, graphs and so on. In this project, we use graphs as the structure to represent the features. Some among many reasons for why graphs are used in the field of computer vision are: the flexibility and powerful representation. The advantages while using this are that there is no fixed dimensionality for the objects and the efficient representation of the relations between the objects and within itself. Last but not the least, graph is an appropriate structure to represent both local and non-local relationships among the object.

The basic problems in computer vision are identification, recognition and classification of objects. Our main goal in this thesis is to recognize and classify the objects. More precisely speaking to build an image retrieval system using different types of features, which are combined together to
form a more reliable feature for this system.

Generally, an image retrieval system consists of describing the image. For decades until today, images are described using textual information in different languages, in this case the image is associated with a set of words, which describes the image. These techniques are widely used in the Internet for example, Google images. These set of words that describes the image are potentially extracted automatically from the web page that contains the involved image, such as the file-name of the image, the page title, etc. The words used to describe the image do not necessarily always reflect the content of the image. In order to overcome this problem, some researchers came up with an image retrieval system where visual data is used to describe an image. This technique uses some pattern recognition methods to extract important features (visual content) that are used to describe the image. This is called content-based image retrieval (CBIR) [4]. This organizes the digital pictures by taking only the visual data into consideration. This approach utilizes computer vision techniques to analyze the image at a pixel level in order to infer the semantic meaning of the scene and to recognize its inner parts including objects and people [5]. This paper is organized as follows: section II describes previous work, section III describes our method, section IV presents the results obtained and section V concludes the paper.

II. PREVIOUS WORK

Object recognition is an active area of research for more than five decades [5]. Many methods have been evolved down the line until today but yet we are not much near, as still there doesn’t exist any method, which give robust results in all conditions and with all images. In order to recognize the object, classification is an intermediate step for the object recognition. Our thesis work is an extension of the previous work [5] where features were extracted using Gestalt theory based on shape, appearance and color distribution. Among all aspects underlying visual information, the shape of an object was used as it plays a special role. They represent the words of the visual language. The main idea of using such descriptor was to represent relevant information about shape, which holds semantic and physical meaning. Besides this, they used them to validate the improvement in recognition when used with others. This descriptor is most common way to describe a shape and include measurements regarding shape properties such as area, perimeter, curvature, junctions, end points, etc. And shape decomposition allows decomposing the shape into simple parts, called primitives and represent some type of topological arrangement of the shape. For their work they used skeleton and its constituent parts. The feature they used was skeleton, which represents the shape of an object. Another feature they used was based on color distribution throughout the object region. They are using this feature to capture the perceptual information associated to the objects appearance, which is a different approach compared to others as they were using only primitive information. Depending on color distribution they defined two types of features, one based on self-similarity measurement and the other constructed by a linear transformation between color pixels and its spatial location inside the objects region. So here they worked with superpixels for self-similarity descriptor. So from their work we are considering to use the three features. Namely Skeleton, Superpixels and Self-similarity.

A. Graph Theory

Normally, a graph represents a set of elements and a set of pairwise relationships between those elements. The set of elements are called vertices or nodes and the relationships are called edges or weights.

A Graph is defined as  
\[ G = (V, E) \]

Where

\[ V = \{v_1, v_2, v_3, ..., v_n\} \] – Vertices,
\[ E = \{e_1, e_2, e_3, ..., e_m\} \] – Edges.

Now, we combine these features with the structure and build the graphs.

III. METHODOLOGY

A. Goals and Method Overview

The goals of our work can be summarized as follows:

- Extract the features according to Gestalt laws.
- Using these features, build different graphs.
- Match these graphs using different graph matching procedures.

B. Graphs and feature representation

In the context of this work, we represent the features extracted using Gestalt theory based on shape, appearance and color distribution using the Graph theory. In Graph theory, there are different types of graphs such as directed graph, undirected graph, weighted graph and so on, which can be used for different purposes. But which graph is more suitable in the context of our thesis in representing the visual data. So after conducting some experiments, we use weighted undirected graph in our project. The reason for using weighted graph is to have weights, which will be very useful when comparing two graphs. The reason for using undirected graph is because here we are not using any flow or path to follow and we need a mutual relation between the pixels.

Now, we build weighted undirected graphs based on the skeleton, superpixels and self-similarity. So, our approach extends these ideas by providing the following:

- Represent the features based on the skeleton, superpixel and self similarity.

1) Skeleton: Skeleton (or medial axis) plays an important role in the body. Sometimes just a glance of the skeleton is enough to recognize the object. It integrates the geometrical and topological features of the object. Now a days, lot of researches are working with skeleton as it is an important shape descriptor for object recognition [6]. Here we consider the Skeleton of an object. By just giving a glance, we can say that it is a Skeleton of Human being that’s the information
we get from a skeleton of an object that’s one of the main reason why researchers are using the skeleton for object recognition. Our motivation is quite same. This gives the basic information of an object.

2) Skeleton graph: Recent years have witnessed a popular way in which skeleton graph is involved in the image comparison problems. Skeleton is the support structure, which is inside, of the body or object.

Skeleton pruning is done after the extraction of skeleton from the object. Then the graph is built using the outcome, which is Skeleton Graph. A skeleton point having only one adjacent point is an endpoint and a skeleton point having two or more adjacent points is a branch point. These points are the vertices and the geodesic distance between the end point and its nearest branch point are the weights between those vertices. The figure 2 is the skeleton graph of an elephant.

However, it is a challenging task to automatically recognize objects using their skeletons due to skeleton sensitivity. Probably the most important challenge for skeleton similarity is the fact that the topological structure of skeleton graphs of similar objects may be completely different. Sometimes the skeletons of the two objects are similar but their skeleton graphs may be very different and vice versa. So summing up, only basing on the skeleton graph object recognition has some false matches. So to make it more robust we also want to include other graphs in our recognition system.

3) Superpixels: Superpixels can be defined as a polygonal part of an image, larger than a normal pixel that is rendered to have the same color and brightness information. In terms of perceptual organization, a superpixel presents more physical meaning regarding the internal structure of the object than a single pixel [5].

4) Superpixel graph: The number of pixels is less, when we consider the superpixels so it is easy to build a graph from the less number of pixels, which will be an advantage for the processing time. As previously we got the structural information of the object now we are looking for some other information, which form the global appearance of the object. So here we consider the superpixels for global appearance. The figure 3 is the Superpixel graph of an elephant.

5) Self-similarity graph: The overall appearance of an object can be described by its internal similar structures. These structures can be used to differentiate the object from other objects, which doesn’t contain such elements. Therefore, [5] devised a more compact self-similarity descriptor based on the work of Irani et al [7]. Superpixels are used by the self-similarity descriptor. The similarity is computed against other superpixels in the object. This self-similarity metric compares each pixel inside a superpixel with the equivalent pixel in the other superpixel. The result was self-similarity surface, a symmetric matrix.

C. Graph matching

Indeed to compare graphs, we perform graph matching in three different types.

1) Pairwise graph matching: Generally, graph matching is done in pairwise i.e. vertices of one graph are matched with vertices of the other graph and the same is done in the case of edges. The figure below shows the concept of
pairwise graph matching [8]. The figure 4 represents pairwise matching.

Here we perform the graph matching of all the three graphs pairwise. Firstly we consider the skeleton graph of one object, which is compared with the skeleton graph of other image. Depending upon the total number of matches and the matches matched we calculate the percentage of matching. In the same way we perform superpixel graph matching and self-similarity graph matching. Then all the three percentages are collected and evaluated and the object recognition process is done.

2) Graphs represented as graph (Graphs in a graph):
We consider the three graphs and represent them as a single graph where vertices are these three graphs and the edges are defined according to the vertices, which they are connected. The below figure 6 shows the fully connected graph.

The adjacency matrix of the graph is shown in the below matrix form ‘G.’

\[
G = \begin{bmatrix}
SK_G & 1 & 1 \\
1 & SP_G & 1 \\
1 & 1 & SS_G
\end{bmatrix}
\]

In the above figure 4.10 and in the matrix, \(SK_G\) represent the Skeleton Graph, \(SP_G\) represents the Superpixel Graph and \(SS_G\) represents the Self-Similarity Graph and the ones represent the full connectivity. And the sample adjacency matrices of the three graphs are shown in the matrix form below \(SK_G\), \(SP_G\), \(SS_G\).

\[
SK_G = \begin{bmatrix}
0 & \cdots & 0 \\
\vdots & \ddots & \vdots \\
1 & \cdots & 1
\end{bmatrix}, \quad
SP_G = \begin{bmatrix}
1 & \cdots & 0 \\
\vdots & \ddots & \vdots \\
1 & \cdots & 1
\end{bmatrix},

SS_G = \begin{bmatrix}
1 & \cdots & 1 \\
\vdots & \ddots & \vdots \\
0 & \cdots & 0
\end{bmatrix}
\]

3) Graph Merging: Generally, in graph merging nodes of the same type are merged together to form a union [9]. The figure 5 shows the concept of graph merging.

Here we are trying to merge the three graphs into one single node, which can be used for matching. The processes is first merge the whole skeleton graph into one single node and do the same for the other two graphs and get the nodes. Then merge all the three nodes into one single node [9]. Now the result is, the three graphs are merged into one single node. This single node is used for image comparison and similar images can be retrieved. This is one way of merging where all the three graphs are merged into one single node. The figure 5 shows the concept of graph merging in this way. The other way is to merge the whole skeleton graph into one single node and do the same for the other two graphs and get the nodes. Now instead of merging all the three nodes into one single node, build a graph using the three nodes. Now the result is, the three graphs are merged into one single graph. This single graph is used for image comparison and similar images can be retrieved. This concept is shown in the figure 6. The superpixel graph and self-similarity graph have a lot of vertices so while merging them into one single node or one
single graph. It is a time consuming process. It may be easy to build a graph using the three nodes than to merge them into one single node. Merging the three nodes into one single node may be quite a difficult task as the three graphs are built using three different features. Generally, Graph Matching is an NPC problem [6], thus, we tried our best to obtain some approximate solutions.

IV. EXPERIMENTS AND RESULTS

A. Data and Experiments

In order to accomplish these goals we have developed all our codes in matlab. In order to test our results we used Caltech 101, which is an image dataset maintained by the Computational Vision group at the California Institute of Technology (Caltech) [10]. The Caltech 101 dataset is considered as a challenging database as the images are taken under different conditions of position, scale, color and illumination which makes Caltech 101 a challenging database with large intra-class variations.

It is important to mention that object detection is not included as part of our solution. We consider that the object is isolated from the background by using the annotation file, which contains information about its contour. We take this information to generate a binary image, which is used by the feature extraction process to concentrate on the object’s region. Due to some inconsistencies found in the annotation files in Caltech 101 we also developed algorithms based on morphology operations to guarantee good contour points as well as to generate smoother and clearer binary images.

In order to evaluate the efficiency of our solution for the purpose of object recognition. We have tested by taking random images from the dataset.

B. Results

1) Skeleton graph: Here we did the image comparison of different images using the skeleton graphs of the images. Then we calculated the matching percentage based on the number of matches matched to the total number of matches. Some of the results for the image comparison are shown in the below table

![Table I](image)

![Image of Table I](image)

The results are good but there are some false negatives when matching is done. This is one of the reason for elephant and plane matching being so high, as the skeleton graphs of both the images are similar. There are sometimes false positives in the some cases such as in the case of cougar. When the elephant is matched to the cougar it gives more percentage, one reason for this could be explained: as the matching depends on the number of nodes present in the image and the global shape. So sometimes there are few chances of false positives. As we can see when the images of elephants are matched they give good results.

2) Superpixel Graph: Here we did the image comparison of different images using superpixel graph. Then we calculated the matching percentage based on the number of matches matched to the total number of matches. Some of the results for the image comparison are shown in the below table

![Table II](image)

![Image of Table II](image)

The results are good but there are some false negatives when matching is done. This is one of the reason for elephant and rooster matching being so high, as the superpixel graphs of both the images at the middle and the neck are similar.

There are sometimes false positives in the some cases such as in the case of cougar. When the elephant is matched to the cougar it gives more percentage, one reason for this could be explained: as the matching depends on the number of nodes present in the image and the global shape. So sometimes there are few chances of false positives. As we can see when the images of elephants are matched they give good results.

V. CONCLUSIONS

In the context of object recognition systems, the general framework goes from considering low-level feature to high-level semantics. During this process, there is gap between the low-level and higher-level semantic. So, Our main idea...
behind this thesis was to bridge the ‘semantic-gap between low-level features and high-level features.

To bridge the gap, we thought of building an image retrieval system in the context of CBIR to retrieve images using image comparison. For this, we represented the features extracted using Gestalt theory based on shape and color distribution using the Graph theory. The main concept was to extract the visual information in the form of object and to represent this object in the form of graph and further use this graph in image comparison for CBIR system.

For this system, we built three graphs using three different types of features. They are Skeleton Graph, Superpixel Graph and Self-Similarity Graph. These are matched in three different ways. Namely, Pairwise Graph Matching, Graphs represented as a graph and then graph matching and the third way is to merge the three graphs. The third way is to merge the graphs in two ways. One way is to merge the three graphs into one single node and then matching and the second way is to merge the three graphs into three nodes and build a graph using them and then graph matching.

This project address and gives an introductory framework to or about graphs which are not still widely in use in the field of computer vision. It explains about different kinds of graphs and how the graphs can be used for the image comparison. The image retrieval system that we are proposing here is a novel system. Previously these features were used but not in the same way as we are using and not in this direction.

REFERENCES

Bibliography: Includes bibliographical references (p. [737]-769) and indexes.
Automatic Detection of the Uterus and Fallopian Tube Junctions in Laparoscopic Images

Prokopetc Kristina
Image Science for Interventional Techniques (ISIT),
UMR 6284 CNRS, Université d’Auvergne, France
Supervised by: Toby Collins and Adrien Bartoli

Abstract—One of the open challenges in medical imaging is to develop automatic methods for interpreting visual information present in endoscopic images during Minimal Invasive Surgery (MIS). A specific objective being currently researched is how to register organs in a pre-operative image such as MRI or CT to endoscopic images. Currently all existing methods for performing this registration problem involve manual input. This includes manually locating the organ and any anatomical landmarks in the endoscopic images. In this paper the organ we focus on is the uterus in gynecological laparoscopic surgery. We proposed the first method for automatically detecting the uterus in a laparoscopic image, automatically segmenting it, and automatically detecting two important landmarks, which are the junctions between the uterus and the two Fallopian tubes.

I. INTRODUCTION

Computer Assisted Intervention (CAI) is the research field which aims to develop algorithms for assisting a surgical team to perform tasks during surgery. One important area of CAI is to perform Augmented Reality (AR) in Minimally Invasive Surgery (MIS). Examples of AR that are being researched today include visualizing sub-surface structures such as tumors and blood vessels captured by a pre-operative Magnetic Resonance Image (MRI) of the patient [3]. An MRI can show the positions of tumors, and so if it can be overlaid onto the endoscopic images, it can show the surgeon where the tumor is during surgery. To solve this problem one has to align (or register) the MRI to the endoscopic image. One way to achieve registration is to detect anatomical landmarks that are visible in pre-operative MRI and laparoscopic images. Once detected their positions can help to solve the registration problem. In this paper we propose a methodology for detecting the uterus and junction between the Fallopian tubes and the uterus in a laparoscopic image based on part-based models [13]. We also present an approach that uses result of the detector to segment the uterus in laparoscopic images.

A. Contributions

The main contributions of this thesis are four-fold:
- The collection of the first database of inter- and intra-patient laparoscopic images of the uterus, which consists of 2838 images annotated with the positions of the uterus and junction locations
- The development of discriminative part-based object detectors for detecting the uterus and tube junctions.
- The development of a contextual model for modeling the geometric relationship between the uterus and the tube junctions, which we learn from training images. This contextual model significantly improves detection performance.
- The development of a method for automatically segmenting the uterus which opens up many future research directions, such as fully automatic registration of the uterus from pre-operative images.

II. BACKGROUND

A. Related Work

Several approaches were recently proposed in [14], [17], [20], [21] to tackle the problem of registering pre-operative images to intra-operative images in image guided medical systems. The authors of [17] propose to use a biomechanical model to compute a volumetric displacement field from partial 3D liver surface motion captured by a stereo laparoscope. This approach was shown to work well for short video clips with manual registration of the CT scan to the laparoscopic images in reference view. A multi-modal image-guided tumour identification approach for robot-assisted laparoscopic nephrectomy was proposed in [14] which uses a semi-automatic method for kidney and tumor segmentation in pre-operative CT scans. It uses an interactive version of the random walker algorithm [2] and requires manual selection of correspondences. Another two similar registration methods for AR - assisted robotic partial nephrectomy were presented in [20], [21]. In the system presented by [20], a single feature on the surface of the kidney was identified and aligned through a spatial translation with the corresponding feature on the surface of the CT scan. Once this point was computed, it was used as a center of rotation around which the image is moved until it coincided with the CT scan. Registration in [21] was achieved using a modified rigid iterative Closest Point (ICP) technique [4], [7], [23].

The methods of detecting or tracking the uterus were presented in [6], [8], [9], [18]. The problem of live image parsing in uterine laparoscopy with the classification of the uterus, surgical tools, specularities and other tissues tackled in [8]. Support Vector Machines (SVMs) were used to learn dense feature descriptors that encoded both color and texture. The patient specific classifier is trained using manually selected Region of Interest (ROI). The system for tracking the uterus proposed in [18] uses a method...
which allows to track the uterus in laparoscopic images by applying a similarity transformation locally using matches between reference frame and a current frame. The authors propose to use Shape-from-Shading to recover the 3D shape of the surface, and the 3D shape is flattened by a conformal mapping which preserves angles on the surface. The objective of the method presented in [6] was to determine which image features belonged to the uterus and to compute their 3D positions. The idea uses the assumptions that the camera is approximately fixed and the image motion is induced only by the movement of the uterus. Therefore features on the uterus can be segmented from features on background structures using the magnitude of their image motion. A way to register the uterus between monocular laparoscopic images in realtime using a two-phase approach was recently proposed in [9]. The objective here was to mask the uterus from the background in a set of reference frames and use the features found within the masked region to reconstruct a 3D model. The authors propose to use Mask Bootstrapping from Motion (MBM) Fig. (27), method which uses a small number of manually-segmented masks to compute the masks in all frames. Once the 3D model is constructed it is used to track the uterus a variant of RANSAC-based rigid pose estimation.

As it was shown mostly all related works for registering pre-operative images to intra-operative endoscopic images use manual input at a starting phase of their methods which include manual selection of correspondences in a set of reference frames. This is very challenging to perform and may take many minutes which introduces delay during surgery. The lack of any camera or organ tracking in [20] means that registration had to be repeated by manually selecting correspondences each time the camera is moved, potentially impacting the surgical workflow. The large inter-patient variation in color, camera and illumination differences between surgeries lead the system in [8] to be patient-specific which restricts its direct application. Moreover it requires user input in order to provide the mask of the tools and uterus for training the color and texture model. Despite of robust feature matching, tracking and augmentation results the system in [18] is not automatic and the surgical site is first marked by the surgeon in the 2D input images and in the reconstructed 3D surface video at an early stage of the laparoscopy. The method from [6] works well when the camera motion is controlled and the uterus motion is significantly greater than the background. However, it cannot work using a single image, and can fail when the camera or background is also in motion. Also if the uterus stops moving then it is possible for features from the background to be wrongly estimated as belonging to the uterus. The methodology proposed in this paper, in contrast, is fully automatic, patient independent and does not require any manual selection. These allows to make the registration process fully automatic by automatically detecting the location of an organ and anatomical landmarks in the laparoscopic images using 2D object detector trained for a specific organ.

B. Part-based Detection

Part-based detectors has been used widely in computer vision. Most of the state-of-the-art detectors [5], [11], [12], [22] show very good results in detecting generic objects in 2D images. To introduce the concepts of part-based detector to medical imaging applications we propose to use the system based on multiscale deformable parts models. In this paper we build our approach using the state-of-the-art part-based detector proposed in [11]. The main idea behind is to represent objects by a collection of parts arranged in a deformable configuration. Each part can capture local appearance properties of an object while the deformable configuration is characterized by parent-child relations between certain pairs of parts. Such representation is well known as pictorial structures [12]. We adapted learning-based system for detecting and localizing objects in images from [15] to detect the uterus on laparoscopic images. The original system is available as open source for study and research purposes. We refer the reader to [11], [16] for detailed description of the methodology and implementation.

III. METHODOLOGY

A. Sources of Class Variability

The abdominal cavity environment in laparoscopic surgery is a very challenging for computer vision tasks. The surgical instruments interacting with the uterus may cause large occlusions, the illumination variations caused by endoscopic light can be very strong, the uterus can undergo bleeding or smokes due to electrocautery, which may disturb organ appearance. We provide a comprehensive list of factors which lead to variation in the appearance of the uterus and Fallopian tube junctions in Table I. We divide them to intra-patient variability factors which means the variability in the appearance of the same uterus and inter-patient variability which appear between different patients and can vary significantly. The variation can be either due to the factors mentioned above also by additional factors due to the fact that the shape of the uterus and the Fallopian tube junctions can be quite different.

<table>
<thead>
<tr>
<th>Sources of Class Variability</th>
<th>Intrapatient</th>
<th>Interpatient</th>
</tr>
</thead>
<tbody>
<tr>
<td>viewpoint change</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>specular reflection</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>smoke</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>bleeding</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>deformation by tools</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>external occlusion</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>resection</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>partial views</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>uterus shape</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>pregnancy stage</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>pathology type</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>anatomic specifications</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>hormonal 'picture'</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

TABLE I: Summarized representation of the possible factors affecting the appearance of the uterus and Fallopian tube junctions in laparoscopic images. We divide this to intra-patient and inter-patients variation.
B. Simplifying the Classification Problem using a Canonical Viewpoint

The sources of class variability are the main sources of the complications for classification problem. There are many possible approaches can be used in order to simplify the detection task and/or adapt the system itself to cope with it. If objects usually appear in a relatively few stable positions with respect to the camera, then they can be represented efficiently in one suitable viewpoint so called canonical viewpoint. The specifications of laparoscopic surgeries such as hysterectomy or myomectomy implies that that the canonical viewpoint of the objects of interest (uterus and Fallopian tubes) is the upright image of the uterus the the laparoscope approximately 10 cm away from it. Additionally, the bladder should appear moved away from uterus so it does not occlude it. The uterus and Fallopian tubes remain in their natural appearance without outer surface deformation cause by tools manipulation (Fig. 1). The definition of canonical viewpoint and usage of the images filtered with respect to this definition allows to overcome the viewpoint change and occlusion variability factors. This simplification also allow us to benefit while constructing training database and its annotation by making the overall annotation task much easier.

Fig. 1: Examples of images with Canonical Viewpoint. The images were rotated and aligned such that they appear in upright position.

C. Proposed Uterus, Junction and Combined Detectors

Our initial idea in the project was to adapt the part-based detector to enforce that two parts in the uterus detector exist at the positions of the Fallopian tube junctions. Thus unlike the greedy selection process for part locations described in [11], we would enforce that two of the parts would be located at, or close to, the positions of the junctions in the training images. To test this idea, we inspected whether two of the parts were naturally being selected at the tube junctions. This would be understandable because the tube junctions contain a lot of discriminative image structure, and more discriminative than e.g. the middle of the uterus or somewhere on the uterus boundary. We found that in some test images the parts did indeed correspond to the tube junctions, but many times they did not. In Fig. 2 we show the positions of the parts of a trained uterus detector in 3 test images. The part locations are marked by blue bounding boxes. In the first image two of the parts have been located at, or approximately at the junctions (Fig. 2a) only the left junction has been located by a part. However the 2b and 2c show other input images where the no parts were located at the junctions. We found that this was occurring in as many as 50% of test images.

Fig. 2: Locations of parts in test images for a uterus detector trained with six parts.

1) Method: Our proposed solution is to train two part-based detectors: one for the uterus and one for the two Fallopian tube junctions. The uterus and junction detectors are part-based detection models trained and run using the methodology and source code provided by [15]. The junction detector therefore has multiple sub-parts, and so it can handle strong geometric variation. The two models are combined with a contextual constraint which constrains the relative position of the uterus and junctions in an image. An overview of the three components of our system is given in Fig. 3. Our detection system works as follows. First the uterus is detected in the input image, and the single best detected bounding box is kept which has the highest detection score that is above a detection threshold. Then the junction detector is run, and all detected junctions are kept if they are above a detection threshold. Finally, we use geometric contextual constraints from the uterus to eliminate false positive junction detections.

Fig. 3: Combined Detector. Contextual constraints defined between uterus and Fallopian tube junctions are shown.

Normally, if the patient does not have any extreme pathology and all the organs, the locations of the junctions are at the left and right sides of the uterus. Depending on the uterus they can appear towards the bottom or towards the top of the uterus body (Fig. 3). Furthermore, the relative size of the junctions compared to the uterus is also constrained. The bounding boxes of the ground-truth uterus and junctions locations are shown in Figs. 3a-3b in white. The center of the bounding box of the uterus is marked with \( q_U \). The centre of the bounding box of the left and right junctions is denoted by \( q_{LJ} \) and \( q_{RL} \) respectively. We show with the number of colored ellipses all the area where the junction most probably may appear due to its anatomical position and relation to the uterus. This schematic representation allows us to think of the task for detecting junctions as the problem of computing the probability of a given position to be a center of a junction (left or right) relative to the center of uterus bounding box. We model this problem as a problem of probability density estimation given a feature vector which describes the spatial...
relationship between the uterus and junctions.

2) **Modeling:** We propose to use Gaussian Mixture Model (GMM) to tackle the task. One of the powerful attributes of the GMM is its ability to form smooth approximations to arbitrarily shaped densities. Given training vectors and a GMM configuration, we wish to estimate the parameters of the GMM, which in some sense best matches the distribution of the training feature vectors. We use the GMMs for a given test image as follows: First the uterus and junction part-based detectors are run on the image. For any uterus bounding boxes, we evaluate the conditional probability of detections from the junction detector using Eq. (2). If the probability is above a threshold $\tau_L$ then we keep it as a positive detection for the left junction, otherwise it is considered as a false positive and the detection is removed. We do exactly the same for the right junctions using a threshold $\tau_R$. To select $\tau_L$ and $\tau_R$ we evaluate the GMM probabilities from the validation sets, and set $\tau_L$ and $\tau_R$ to be the 98th percentile (i.e. the probability for 98% of the validation examples is $\tau_L$ and $\tau_R$ respectively).

3) **Testing:** We use the GMMs for a given test image as follows: First the uterus and junction part-based detectors are run on the image. For any uterus bounding boxes, we evaluate the conditional probability of detections from the junction detector using Eq. (2). If the probability is above a threshold $\tau_L$ then we keep it as a positive detection for the left junction, otherwise it is considered as a false positive and the detection is removed. We do exactly the same for the right junctions using a threshold $\tau_R$. To select $\tau_L$ and $\tau_R$ we evaluate the GMM probabilities from the validation sets, and set $\tau_L$ and $\tau_R$ to be the 98th percentile (i.e. the probability for 98% of the validation examples is $\tau_L$ and $\tau_R$ respectively).

D. **Using the Uterus Detector to Automatically Segment the Uterus in Laparoscopic Images**

We have seen in **II-A** that the task of segmenting the uterus in laparoscopic images is an important component for existing AR systems. It has been shown that segmentation is performed manually by demarking the uterus in one or more images. Our approach is based on the fact that it is possible to segment the uterus automatically from background structures if we have patient-specific texture and color information for the uterus and background structures a priori [8]. We propose using the bounding box provided by our uterus detector to automatically obtain the necessary texture and color information in a patient-independent manner. Once obtained, we then segment the uterus in the input image by classifying each pixel using a similar approach as [8]. Our approach can therefore be thought of obtaining a rough segmentation of the uterus using an patient-independent method (i.e. the patient-independent object detector), which is then followed by learning a patient-specific model of the
uteros’ texture and color. This patient specific information is then used to provide an accurate segmentation. In Fig. [5] we present the proposed pipeline for automatically constructing the patient-specific classifier using the output from the uterus detector.

![Proposed processing pipeline based on [8] for automatically constructing a patient-specific classifier for segmenting the uterus using the output from the uterus detector.](image)

Fig. 5: Proposed processing pipeline based on [8] for automatically constructing a patient-specific classifier for segmenting the uterus using the output from the uterus detector.

Our method works as follows. First the uterus detector is run on the input image. If there are any positive detections then we keep the single bounding box with the highest detection score. We then make the assumption that within the bounding box the shape of the uterus is approximately an ellipse centered at the center of the bounding box and with its axes oriented with the x and y-axes of the image. We are safe in making the assumption that its axes is approximately aligned in this way because the images used to train the uterus detector were rotated so that the line passing through the two Fallopian tube junctions is parallel to the y-axis. Because the uterus has approximate bilateral symmetry, and when viewed from the canonical viewpoint gives an approximately elliptical image, then the ellipse axes should approximately align with the image axes. Denoting the bounding box center, height and width by \( p \in \mathbb{R}^2, h \in \mathbb{R} \) and \( w \in \mathbb{R} \) the ellipse is given by:

\[
\frac{1}{w} (x - p_x)^2 + \frac{1}{h} (y - p_y)^2 = 1
\]  

(3)

We then assume that at a small region of the ellipse centred at \( p \), if the detection is a true positive, then we can be very sure that pixels within this region belong to the uterus. By the same token, for all pixels that lie very far outside of the the ellipse then we can be very sure that these do not lie on the uterus and so can be considered as background pixels. For all other pixels, we cannot be certain whether they lie on the uterus or on the background, and so these need to be classified with another means. Specifically, we classify these using texture and colour models built from the pixels that have been labelled as uterus and background with high certainty. We therefore make an initial classification of each pixel \((x, y)\) in the image into three types: either uterus, background or uncertain. This is given by the following rule:

\[
\begin{align*}
 f(x, y) < 1 + d & \Rightarrow (x, y) \text{ is classed uterus} \\
 f(x, y) > 1 - d & \Rightarrow (x, y) \text{ is classed background} \\
 1 - d \leq f(x, y) \leq 1 + d & \Rightarrow (x, y) \text{ is uncertain} \\
 f(x, y) = \frac{1}{w} (x - p_x)^2 + \frac{1}{h} (y - p_y)^2 & \geq 0 \leq 1
\end{align*}
\]

(4)

The free parameter \( d \) denotes how conservatively we classify pixels using the ellipse. When \( d \) is large then only pixels within a small region at the center of the ellipse are classed as uterus, and only pixels very far outside of the ellipse are classed as background. Although a conservative value of \( d \) is advantageous to prevent mis-classification of pixels, if it is too large then we may not be able to obtain a sufficient number of pixels with which to model well the color and texture characteristics of the uterus and background. In our experiments we have found a value of \( d = 0.2 \) works well and meets a good trade-off, and in all training examples \( d = 0.2 \) never causes background pixels to be incorrectly classified as uterus.

We then train a patient specific texture and colour classifier using all pixels labeled as background and uterus with the approach of [8]. Once trained, we then apply this to classify all unclassified pixels. We achieve this using the same models as developed in [8]. The difference between [8] and our approach is that we are running a patient-specific segmenter on the same input image as was used to train the texture and color models. After the segmenter has classified all unclassified pixels, the segmentation is cleaned up with several post-processing steps as described in [8] to remove holes and retain only the largest uterus segment.

E. Constructing the Training Database

At present there exists no publicly available collection of laparoscopic data that might serve as a training database for this project. Here we present the first database of laparoscopic images of the uterus. The constructed database was annotated with respect to the format used in PASCAL dataset [10]. The data for database database was obtained from two different sources. The primary source was hospital CHU Estaing in Clermont Ferrand. The dataset collected from the hospital consists of videos from 4 of gynecologic laparoscopy. A videos were captured by a Karl Storz laparoscopy system and the size of image at each frame is 1048x576 pixels at 25 fps. Each captured video file was converted to the image sequence. The data collected from internet was used as the second source with respect to the doctrine of ‘fair use’ [1]. It includes still images of uterus as well as images obtained from videos of laparoscopic surgeries. All images containing 80% visible uterus body with
Fallopian tubes in a canonical viewpoint as defined in section III-B were considered as positive examples. Other images, where the uterus body was no present were considered as negative examples. (Table II)

<table>
<thead>
<tr>
<th></th>
<th>train</th>
<th>val</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>uterus</td>
<td>382</td>
<td>804</td>
<td>1186</td>
</tr>
<tr>
<td>junction</td>
<td>1037</td>
<td>1060</td>
<td>2097</td>
</tr>
</tbody>
</table>

U 1 part(s) (AR = 0.514) (ANFPPI = 34)
U 2 part(s) (AR = 0.536) (ANFPPI = 13)
U 4 part(s) (AR = 0.503) (ANFPPI = 15)
U 6 part(s) (AR = 0.517) (ANFPPI = 14)
U 8 part(s) (AR = 0.481) (ANFPPI = 19)
U 10 part(s) (AR = 0.536) (ANFPPI = 13)
U 12 part(s) (AR = 0.589) (ANFPPI = 10)

Junctions. The detection results from uterus detector are shown in 9a, 7d. The results from junction detector presented in 9a, 7e and combined detector results are in 9c, 7f. This illustrates how the false positives produced by junction detector were eliminated using contextual constraints from the uterus.

To evaluate the segmentation performance we used all positive uterus test images. For each of these images we manually computed a ground-truth segmentation. We evaluated performance using the DICE score. Because our method is the first method which can fully automatically segment the uterus, we though a reasonable baseline method would be the segmentation obtained using the bounding box of the uterus detector. In Fig. 8 we show the performance of the...
that image. In Fig. 9 we show some example segmentations.

![Dice boxplots showing performance improvement of our proposed segmentation system](image)

**Fig. 8:** Dice boxplots showing performance improvement of our proposed segmentation system.

![Examples of automatically segmented uteri using our proposed method](image)

**Fig. 9:** Examples of automatically segmented uteri using our proposed method.

V. CONCLUSIONS AND FUTURE WORKS

The initial goal of this thesis was to develop a method for detecting the uterus and junctions between the Fallopian tubes and the uterus in a laparoscopic image based on part-based models [13]. After reviewing recent related work and the state-of-the-art it become clear that the Object Detection with Discriminatively Trained Part Based Models [11] will provide the strong and efficient base to our methodology. The detection method described in this paper is fully automatic, patient-independent and does not require any manual selection. This allows to make the registration process fast and automated. The remarkable performance of the uterus detector encouraged us to propose our automatic uterus segmentation approach. The segmentation results are very encouraging. Although the presented work satisfies the initial goal of the project, we have defined a number of directions for future research. First of all we are looking forward to improve on the junction detector by using more specialised features than PCA-HOG features in the part-based detector. One possible direction would be to include the texture features that were used in [8]. We also aim to investigate online learning to adapt the detectors to different viewpoints of the uterus. We also want to use the uterus and junction detectors to enable fully automatic registration of a pre-operative MRI of the uterus to intra-operative laparoscopic images. We want to incorporate our automatic method for uterus segmentation into the method of [9] to allow fully-automatic Structure-from-Motion of the uterus from laparoscopic images.

VI. ACKNOWLEDGMENTS

This research has been supported by ERC Starting Grant GA307483 through the project of excellence FLEXABLE (Deformable Multiple-View Geometry and 3D Reconstruction, with Application to Minimally Invasive Surgery).

REFERENCES


Automatic spatial and temporal organization of long range video sequences from low level motion features

Alberto Quintero Delgado, Yannick Benezeth, Déside Sidibé
Université de Bourgogne
ajquinterod@gmail.com, {yannick.benezeth, dro-desire.sidibe}@u-bourgogne.fr

Abstract—In this paper, we address the analysis of activities from long range video sequences. We present a method to automatically extract high level scene semantics from low level motion features. Literature on the area propose the use of complex probabilistic models, here instead, we propose a much simpler method that nevertheless obtain high level semantics and global scene states. The scene layout is first extracted, with a set of regions that have homogeneous activities called Motion Patterns. These regions are then analyzed and the recurrent temporal motifs are extracted for each Motion Patterns. Through experimentation we analyze the use of several proposed video representations and motif recovering approaches. Results show that our method can accurately extract important temporal motifs from video surveillance sequences.

I. INTRODUCTION

Most of public areas are now monitored with surveillance cameras. The manual analysis of all this information is clearly impossible. Nowadays, most of these CCTV are never screened and are recorded to be later used as evidence. However, even the search for specific events in a large collection of video is also a very time consuming and tedious task that can be impossible if the amount of video is too large. It is conceivable to browse several hours or days of videos if there exists some metadata about the content of the video.

Therefore, an automatic way of processing all the recorded data to compact the videos, classify and browse this data is needed. But, because video scene understanding is not an easy task it has kept the researchers on the field very busy.

Moreover, if we consider that not all the videos are the same, for instance, one video can has really dense activity flow and another can be very sparse. Also, one video can has structured and simple layout and another can has very complex with overlapping activity areas. In the same way we can find videos with structured activity patterns and another with random activities. All this makes very difficult to come up with an unique solution without considering a lot of constrains or making a lot of assumptions.

Being able to analyze the scene, to automatically identify normal and abnormal behaviors could be really useful. If we consider a method that can identify abnormal activities could be possible with this to highlight or to show the streaming from that camera in a big main screen.

Activity mining in video analysis can be used not only for a later browsing or classification of the videos, but many applications can be achieved. Two main task can be considered in the video analysis area; to recover the spatial organization in a video scene and to recover the activities happening in a video sequence.

Retrieving the spatial organization in all the scenes of those videos could ease the activity mining task, and also to understand a classify videos, that is why, we also address this in our approach. Manually extract the activity patterns in video sequences is not an easy task. If we consider for instance a network of surveillance cameras that record all day long, this task would be very time consuming, almost impossible.

Many video content analysis methods use conventional tracking-based methods to recover individual moving object trajectories [7]. Specific information about moving objects such as position, velocity or acceleration is then used for high-level tasks. However, it is broadly accepted that object tracking is ill-suited for videos with a large number of moving objects such as in crowded scenes. Other video representation have been proposed in the litterature based on background subtraction [1], optical flow [4] or spatio-temporal features [6].

These low-level features are latter used with supervised or unsupervised learning methods to recognize a specific activity or identify an usual behavior. Supervised approaches can perform quite well when accurate labels are provided however it can be difficult to explicitly define positive and negative classes in complex scenes when several activities occur simultaneously. To take into account the sequential nature of activities and the possible causal dependencies between them, it is necessary to use more complex models such as Hidden Markov Model [5], Bayesian network [3] or Probabilistic Topic Models [8].

II. METHODOLOGY

Our proposed approach is based on three steps: 1) computing low level motion features, 2) extracting the spatial organization of the scene (Motion Patterns), and finally, 3) using the obtained MPs to extract the recurrent activity patterns or motifs in each of them. Fig 1 show a diagram of our approach.

A. Motion features

As part of the first stage of our approach, we take as input long range video sequence that lasts 1 hour. The number of
frames in each sequence is very high and in order to reduce the number of operations and hence, the computational time, all frames are resized to 128x128.

The next step is to compute the low level motion features for every pair of frames in the whole video sequence. We compute dense optical flow using the method developed in [2]. The flow matrix is then divided into a grid and just one value is kept in each of its cells:

\[
C_i(x_c, y_c) = \begin{cases} \theta, & \text{if } \bar{\rho} \geq \rho_t - 1, \\ -1, & \text{otherwise} \end{cases}
\]

where \(C_i\) is a cell in the grid \(G\), \(x_c\) and \(y_c\) are the coordinates of the center of \(C_i\), \(\theta\) is the dominant angle in that cell, \(\bar{\rho}\) is the average magnitude of all the flow vectors in \(C_i\) and \(\rho_t\) is a threshold.

\[
S_1 = \begin{pmatrix} d_{1,1} & \cdots & d_{1,m} \\ \vdots & \ddots & \vdots \\ d_{m,1} & \cdots & d_{m,m} \end{pmatrix}
\]

where \(m\) is the number of cells and \(d_{i,j}\) is a distance function that measure the similarity between the histograms \(\Phi_i\) and \(\Phi_j\). In this case we use Euclidean distance because the representation used are Histograms.

Later, in a next step of our approach, we compute the Symmetric Normalized Laplacian matrix \(L\) given in (3) from the similarity matrix \(S\) using the diagonal degree matrix \(D_{i,j} = \sum_j S_{i,j}\).

\[
L = I - D^{-1/2}SD^{-1/2}
\]

where \(I\) is the identity matrix. To this matrix \(L\) we do
an eigendecomposition such that \( L = Q\Lambda Q^{-1} \) where \( Q \) is an square matrix and whose columns represent each of the eigenvectors of \( L \) and \( \Lambda \) is a diagonal matrix whose diagonal elements represent each of the eigenvalue of \( L \) such that \( \Lambda_{i,i} = \lambda_i \).

In order to find spatial organization of activities in the video, a hierarchical clustering is applied on the symmetric normalized Laplacian matrix \( L \).

We set the number of clusters so that 90% of the total variance is retained, i.e.

\[
\sum_{i=1}^{k} \lambda_i \geq \tau_2, \tag{4}
\]

being \( \lambda_i \) the eigenvalues in \( \Lambda \), \( \tau_2 \) a defined threshold (in this case 90%) and finally \( k \) is going to be the maximum number of clusters.

The obtained clusters define the Motion Patterns, and all the cells belonging to the same class are said to be part of the Motion Pattern or MP represented by that class. Figure 4b shows an example of the results after the clustering. As it can be seen in the resulting image there are some isolated cells. To clean the image there are some isolated cells. To clean the resulting image there are some isolated cells. To clean the image a common filtering is applied, using the majority or the most common value for every cell using its 8 neighbors. This is done in cell-based resolution not in pixel-based resolution, and the results are shown in Figure 4c.

From the matrix \( F \) we compute the distance between each pair of video clips obtaining the similarity matrix:

\[
S_2 = \begin{pmatrix}
\delta_{1,1} & \cdots & \delta_{1,m} \\
\vdots & \ddots & \vdots \\
\delta_{m,1} & \cdots & \delta_{m,m}
\end{pmatrix}
\tag{6}
\]

where \( m \) is the number of video clips and \( \delta_{i,j} \) is the Cosine distance between video clips \([SC_{1,i} \cdots SC_{m,i}]^T\) and \([SC_{1,j} \cdots SC_{m,j}]^T\). Here is important to mention that we use the Cosine distance because we are measuring similarities between angles, this is because we are using the FullSeq representation. Also, it is important to mention that because we convert the orientation angle \( \theta \) to the unit circle coordinates we check all the incidences of \(-1\) (no-activity) and assign \((0,0)\) so the distance from no-activity to any other activity or orientation is always the same.

After getting the labels for each small video clip we obtain a signal representing the long video sequence and applying a majority vote filter to remove noise we end up with a long signal that we can represent as a string \( S \).

In a following step we remove all the repetition of every single activity on \( S \), for instance the string "AABBCC" would become "ABC". We pass the string \( S \) as parameter to the Algorithm 1, which returns a list of tuples. Each entry of this list is of the form ("ABC", 20), where "ABC" is the substring found in the original signal and 20 is the number of occurrences of this substring.

**Algorithm 1** Substrings recovering algorithm  

**Require:** Long string filtered representation of video \( S' \)  

**Ensure:** subs as set of tuples (motif, occurrences)

1: \( subs \leftarrow \emptyset \) \( \triangleright \) Empty set for future results  
2: \( init \leftarrow 0 \) \( \triangleright \) The first position of \( S \)  
3: \( end \leftarrow length(S) \)  
4: \( \varphi \leftarrow S[init] \) \( \triangleright \) The first character of \( S \)  
5: for \( i \leftarrow init \ (+1) \) to end do  
6: \( rest \leftarrow S[i : end] \) \( \triangleright \) All characters from \( i \) to end  
7: for \( j \in rest \) do  
8: \( \varphi \leftarrow \varphi + j \)  
9: if \( \varphi \notin subs \) and \( S.count(\varphi) < 2 \) then  
10: \( nlen \leftarrow length(\varphi) \)  
11: \( \varphi' \leftarrow \varphi[0 : nlen] \)  
12: if \( length(\varphi') > 1 \) then  
13: \( tmp\_tuple \leftarrow (\varphi', S.count(\varphi')) \)  
14: \( subs.add(tmp\_tuple) \)  
15: end if  
16: \( \varphi \leftarrow S[i] \)  
17: break  
18: end if  
19: end for  
20: end for  

The returned list is sorted by the number occurrences of the substrings in decreasing order, which mean that the first entry of the list is the substring that repeat the most (without considering its length). Having this list we can easily use any
criterion to retrieve our motif; our approach is to get the most recurrent substring (MSR) in disregard of its length, and as result we finally obtain the most recurrent substring which in this case is our recurrent motif.

III. EXPERIMENTS

We selected three video sequences for running our experiments on them. For each of the video sequence the motion patterns ground truth was manually extracted for a later qualitative comparison with the output of the first stage of our approach. The three video sequences selected are surveillance video from street traffic where the activity flow is controlled by traffic lights. Each of the video has a minimum length of 50 minutes. The Figures 6, 7 and 8 shows the first frame and an example of the manually extracted ground truth motion pattern masks for the three videos: Hospedales_1, Hospedales_3 and Kuettel_4 respectively.

Fig. 6: First frame + example of ground truth motion patterns for Hospedales_1

Fig. 7: First frame + example of ground truth motion patterns for Hospedales_3

Fig. 8: First frame + example of ground truth motion patterns for Kuettel_4

From the output of the Algorithm 1 we get the most recurrent substring (MRS) which we treat as the most common motif, in this case we keep the MRS without considering its length. We propose three approaches for doing the match of the recovered motif and the original signal.

- **Exact motif matching**
  For this approach what we propose is to do a general matching only on those occurrences of the recovered motif not allowing any change of variation of the motif. In this case we do not consider the duration of each of the activities in the motif.
  To explain how the general matching is done we are going to go through an example; For instance, if the recovered motif is $m' = ACB$ and the original signal is $s' = AAAAAACCCCCBBB$, the entire signal $s'$ match the motif $m'$ because the recovered motif is only the pattern of the activities in without considering each activity’s duration.
  On the other hand, if we have the signal $s'' = AAACCCBBBBEEEEE$, the same motif $m'$ will only match from the first occurrence of the activity $A$ to the last occurrence of the activity $B$ leaving out all the occurrences of $E$.

- **Partial motif matching**
  This approach works as the previous one (the exact matching) but with the difference that now we allow one of the activities of the motif to be replaced by another one. The replacement could be only one at a time, for instance, if the recovered motif is $ABC$ our approach will match every occurrence of $ACB$ and also $BCB$.
  For this we create a list of possible partial motif. Like this, if the number of clusters was for instance 3, that means that the possible activities are $A$, $B$ and $C$. Hence, the list of possible partial motif for $ACB$ will be $[BCB, CCB, AAB, ABB, ACA, ACC]$. Figure 5 show a plot of the signal obtained after clustering the small video clips with a motif partial matching with different color for each occurrence.

- **Average motif matching**
  Finally for the last approach, it is needed to loop over the entire signal first, for performing a exact matching and after this, the mean $\mu$ and the deviation $\sigma$ of the duration of every match is computed. In a later step these matches are filtered and we only keep those matched whose duration fall in the range of $[\mu - \frac{\sigma}{2}, \mu + \frac{\sigma}{2}]$.
  Also, for the experiments that we ran, we considered three evaluation criterion. The three of them give quantitative results and lead us to a better configuration of all the thresholds, data representation, filtering and clustering techniques.
  All the criterion are based on the matching of the recovered motif on the original signal.

(a) **Amount of signal matched**
  With this criterion we compute the ratio between the amount of signal matched and the length of the original signal. It is important to mention that when we state “original signal” we mean the long string sequence that represent the long video sequence and that comes as output from the clustering step.
  After the motif matching we are able to know which
section of the original signal was matched or not, and computing this ration give a quantitative value that is close to 1 for a complete matching of the original signal or close to 0 for a poor matching. In Figure 9 we show an example of a periodic signal (the perfect video sequence scenario) and the recovered motif matched (in different colors for each occurrence), in this case the amount of signal matched ratio would be 1.

Fig. 9: Example of periodic signal with motif matched highlighted (see color version).

(b) Recovered Periods
This criterion gives the ratio between the expected number of periods and the actual number of periods matched with the recovered motif. First, we compute the expected number of periods and we do this by computing the Fast Fourier Transform of the original signal. In a later step, just dividing the length of the original signal over the found period we obtain the expected number of periods. After this, because we also know the number of times the recovered motif is matched in our original signal (for instance, this number of occurrences can be checked in Figure 9) we computer the ratio between the matched periods (the occurrences of the motif) and the expected number of periods.

(c) Period Length
The final criterion proposed to measure the accuracy of our approach is also a quantitative value, the ratio between the length or duration of the average motif and the expected period found in (a).

In the same way, the thresholds throughout these experiments and all the other experiments during the realization of this project were setted as follow: \( \tau_1 = 0.06 \) and \( \tau_2 = 0.90 \) for the spatial organization and \( \tau_2 = 0.95 \) for the temporal organization.

From these experiments we could found what feature representation works better for spatial organization and which one works better for temporal organization, we could also found the best setting for our thresholds.

In Fig 10 it can be seen the recovered spatial distribution (motion patterns) from the selected videos.

IV. DISCUSSION

After analyzing the results (experimental section) of our experiments and its outputs we want to use this section to discuss of at least one possible application for our method. For instance, one of its application could be to improve already existing tracking techniques.

If we consider for example a method like Particle Filter in which we have a model of the object been tracked and generate randomly certain number of particles around the last tracked position and then we do a weighted comparison between the model of the object with the new model at each of the generated particles and like this we can obtain the new position of our object.

Now, if we also consider the spatio-temporal organization of our scene, where we know the position of the tracked object (this tell us to which motion patterns it belongs), at a time \( t \) we know which activity (the classes from our clustering stage) is more probably to happens and with this we can put more emphasis on that region for the generation of particles for instance.

V. CONCLUSIONS AND FUTURE WORKS

A. Conclusions

After reviewing the literature on video analysis, we have proposed a method that taking a long video sequence (making the assumption that is has some activity pattern) it can recover the spatial and temporal organization (Motion Patterns and Motifs respectively).

Throughout all the experiments done we were able to verify that the use of histograms for the features representation yields to better results when recovering the spatial structure of the scene. Also, a more detailed representation as the Full Sequence proposed yields to better results when recovering the activity patterns on the video sequence.

Our proposed method simplify the activity patterns recovering task by first finding the spatial structure of the scene, allowing us later to use a simple string matching approach instead of having to build complex models.
In the same way, we were able to verify that it is very difficult to retrieve the activity patterns on video sequences when the activity flow is sparse with this kind of methods. On the other hand, this approach shows promising results on videos with a dense activity flow.

B. Future Works

As we could verify, with our proposed approach it is difficult to retrieve the motion patterns on video sequence when there is an sparse activity flow, hence, as future work we have proposed to find an approach that can handle this kind of situations. Perhaps, a different feature representation combined with a different distance functions.

Also, we have proposed to try a different approach from signal analysis to recover partial periodic components on the Full Sequence signal representation.

In the same way, we have proposed to improve the method to recovering the spatial organization on video scenes when there are overlapping motion patterns.

And finally, we are going to implement the improved particle filter tracking approach using the output from our method.

REFERENCES