2012 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp 65-70

Object Pose Estimation and Tracking by Fusing Visual and Tactile Information

Joao Bimbo[1], Silvia Rodríguez-Jiménez,[2], Hongbin Liu[1], Xiaojing Song[1], Nicolas Burrus[2]
Lakmal D. Senerivatne[1], Mohamed Abderrahim[2] and Kaspar Althoefer[1]

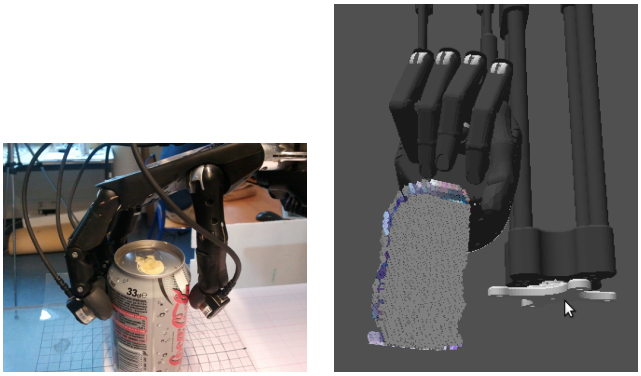
***Abstract*— Robot grasping and manipulation require very accurate knowledge of the object's location within the robotic hand. By itself, a vision system cannot provide very precise and robust pose tracking due to occlusions or hardware limitations. This paper presents a method to estimate a grasped object's 6D pose by fusing sensor data from vision, tactile sensors and joint encoders. Given an initial pose acquired by the vision system and the contact locations on the fingertips, an iterative process optimises the estimation of the object pose by finding a transformation that fits the grasped object to the finger tips. Experiments were carried out in both simulation and a real system consisting of a Shadow arm and hand with ATI Force/Torque sensors instrumented on the fingertips and a Microsoft Kinect camera. In order to make the method suitable for real-time applications, the performance of the algorithm was investigated in terms of speed and accuracy of convergence.**

I. INTRODUCTION

"The whole is greater than the sum of the parts" is a well-known aphorism stemming from the theory of Gestalt psychology. Its principle is that we, humans, perceive the outside world in an organized, holistic way, capturing an object's entire essence before being aware of the individual parts that compose it. If this notion is translated to the field of robotics then it can suggest that interpreting a robot's sensory data as a whole, fusing data from different sensors together can provide richer information than when taking that same data separately.

This fusion of information enables a robotic system to better perceive the environment and monitor a task's progress, creating synergies that can provide redundancy and increase the system's robustness [1]. In the field of robotic grasping and manipulation, two types of sensory data stand out as key for the successful completion of an autonomous task: vision and tactile sensing [2]. The ability to recognise an object, accurately locate it and continually track its 6D pose – position and orientation – is of fundamental importance in robotic grasping.

Given that an object's pose acquired from the vision system may not be accurate due to limitations of the camera's depth information, bad calibration or occlusions, one can rely on the robot's sense of touch to increase the precision of the pose estimation [3], [4]. This paper details the data acquisition from the vision and tactile sensors and presents a method to accurately estimate the pose of an object by obtaining its 3D shape and an initial pose from the vision system and then finding a transform that rectifies its pose by fitting it to the fingers' current contact locations, obtained through the computation of the forward kinematics and tactile sensors on the finger tips. The required 3D transform is found using the Levenberg-Marquardt (LM) iterative method and is one that modifies the object's pose so that the contact locations on the fingers coincide with the object's surface. In order to be able to use this method in a real-time application, rectifying the object's pose whenever it is required by the robotic system, computational speed becomes an issue along with the precision of the algorithm. Fig. 1(b) shows an example where, due to an inaccurate calibration between the camera and the robot, the fingers do not seem to touch the object, even though we know from the information from the tactile sensors that the fingers are actually in contact, as it can be seen in Fig. 1(a).



(a) Fingertips touching object (b) Robot and object models

Fig. 1. Initial pose estimate as given by the vision system

This paper details related work in Section II, introduces the data acquisition strategy in Section III and describes the pose fitting method in Section IV. Results using both simulated and real data are presented in V and the discussion on these results and future research directions in the field are elaborated in Section VI.

II. RELATED WORK

A wide number of researchers have dealt with sensor fusion, particularly the synergistic coupling of vision and tactile sensing. Luo *et al* [1] elaborated a comprehensive overview

[1]J. Bimbo, H. Liu, X. Song, L. Senerivatne and K. Althoefer are with the Centre for Robotics Research, King's College London, WC2R 2LS London, United Kingdom, {joao.bimbo, hongbin.liu,song.xiaojing,lakmal.senerivatne k.althoefer} @kcl.ac.uk

[2]S. Rodriguez, N. Burrus and M. Abderrahim are with the Department of Systems Engineering and Automation, Carlos III University of Madrid, Leganés, Spain, srjimene, nburrus, mohamed@ing.uc3m.es

of different multi-sensor fusion strategies and presents examples of applications in different areas of robotics.

There are many techniques for tracking objects using vision, but they can be classified into three groups according to object properties: feature-based, model-based and optical flow-based approaches. In the first category, the most popular ones are particle filter, Kalmar filtering, Scale Invariant Feature Transform (SIFT), mean shift, Multiple Hypothesis, among others [5]. However, once the robotic hand is enclosing its fingers around the target object and when the object is grasped, the fingers occlude the object and the visual tracking could fail.

Since the field of robot grasping and manipulation requires different types of sensor information and demands a high level of accuracy and robustness, sensor fusion has been extensively applied to deliver better results. Allen *et al* [2], [6] describes a system that integrated vision and local contact/force information and conducted a number of experiments to validate the proposed design, Prats *et al* [4] used a combined vision with tactile and force information to perform the task of finding a door handle and opening a cabinet. The sensory data were used as input in a control loop and the performance under different scenarios was evaluated and compared: using only force feedback, using vision and force and finally using the combination of tactile, vision and force. This last setup proved to perform better than using only force or a vision-force combination. Petrovskaya *et al* [7] used a probabilistic approach to estimate an object's localisation using tactile exploration. Hebert *et al* [3] developed a method to estimate a grasped object's pose by combining stereo vision with a force/torque sensor mounted on the wrist of a robotic hand. Using also the information from the joint sensors the proposed method was able to infer the contact modes (a mapping of fingers to each of the object's faces) and accurately estimate the object's location. The work done by Honda *et al*[8] is closely related with the one described on this paper but is based on different assumptions such as the shape of the object requiring to be composed of plain and quadratic surfaces and the vision estimate to be very accurate.

III. MULTI-MODAL SENSING ARCHITECTURE

The presented multi-modal perception robotic system assumes the following setup:

- A Microsoft Kinect camera [9] which observes the grasping scene. It is mounted on the robotic arm.
- 6-axis force and torque sensors instrumented on the fingertips of a multi-fingered robotic hand.

The multi-modal system and its components should be calibrated in order to represent the target object in a reference coordinate system. The transformation from the robot base to the camera is estimated through OpenCVs checkerboard corner [10] detector using a single pose. The calibration is based on detecting a planar checkerboard pattern, with its position with regards to the robot base being known. Although the calibration is refined using the depth image, an error in the range of centimeters will persist due to the precision of the Kinect, the use of a single checkerboard pose and possible inaccuracies in the kinematics model of the robot.

The proposed fusion of vision and tactile sensing allows making accurate measurements of grasped objects in spite of the limitations and inaccuracies in calibration.

A. Vision

The vision system provides the 3D shape and the initial pose of the target object. In this paper, we assume textured-enough objects, of which a 3D model has been previously acquired and stored in a database. If the model of the object is not already stored in the database, an acquisition process is used to reconstruct it. The method, summarized below, has been developed [11] to cover online scenarios where the robotic system has to grasp and manipulate new objects (previously unknown).

1) Object Model: The acquisition of the 3D models has been done using a single RGB + Depth image of objects lying on a table. This method allows a fast reconstruction of unknown objects and the acquired models are precise enough to compute reliable grasping points. The Kinect, which is oriented to get a top view of the objects, only provides the geometry of the visible parts. Thus, we estimate the hidden parts by exploiting the geometrical properties of everyday objects, which can often be approximated as the result of an extrusion process. In the first stage, a table-top object detector identifies and extracts a cluster of 3D points belonging to the object. Then, existing points are extruded along the table plane normal to fill a voxelized volume around the cluster of interest. Object concavities may get filled during the extrusion step, which we compensate by checking the voxel consistency against the depth image. The object boundaries are finally refined using color segmentation and then missing depth values are filled using image inpainting. Once the point cloud of the object is obtained, it is stored in a database along with its color view for the recognition stage.

2) Recognition and pose estimation: For recognition and pose estimation, when the textured object is being grasped, if some part of the object is still visible, the most popular techniques available in the literature are based on local features [12]. This is because of its robustness to partial occlusions as long as enough features can be matched in the visible part.

In this paper, a method derived from the proposed by Lowe *et al* [13] is used for recognition. To obtain the pose estimation from few feature matches, the depth information is used to reduce the influence from background and occlusions using a 3D reprojection error[14].

For getting the best match, a feature matching is computed between the SIFT points extracted from the image under analysis and the SIFT points extracted from each color image corresponding to the reference models. The best matches are the closest ones. Each single feature point association results in a candidate 2D transformation for its corresponding view. Once the object has been recognized, its 6D pose is estimated

using RANSAC to discard outliers. We rely on iterative least square only to minimize the reprojection error using 2D feature coordinates and introducing the depth information. Thus, a 3D reprojection error is computed taking into account the 3D coordinates $H(F_i)$ of the model feature i projected on the current image and the coordinates $C_i(x, y, z)$ of the detected feature point in the image under analysis.

$$err = \sum_{i=0}^{n} \Delta(H(F_i), C_i(x, y, z))^2. \quad (1)$$

The pose is provided taking into account the transformation between the recognized object and the reference object stored in the database. The transform is given as a translation vector and a rotation quaternion.

B. Tactile Sensing

The second source of information came from the robot's sense of touch and it is commonly referred to as "intrinsic contact sensing". In this paper we adopt a tactile sensor developed and validated in previous work [15]. This method uses a 6-axis force and torque sensor mounted on the fingertip to precisely estimate, among other information, the location of the contact. By attaching the sensor under a parametrisable convex surface, one can find the coordinates of a point on that surface that satisfies the equations that describe the acting forces and moments [16]. Besides this contact location coordinates, an extra parameter is calculated which relates with the local torque acting along the normal direction on that point.

The accuracy and robustness of this sensor has been verified to have a root mean square error of 266 μm, making it suitable for use in robot grasping and manipulation tasks and the frequency of data acquisition was set to 100 Hz [15].

IV. POSE-FITTING METHOD

A. Description

This method takes as inputs a point cloud of the object and the location of each finger that is in contact. It requires at least two fingers to touch the object and increases the accuracy of its result with the number of independent contacts.

The first step of the algorithm consists of transforming all the contact locations and the object point cloud into the same reference frame. The choice of a frame on the robot hand's palm when tracking a grasped object, presents advantages such as, during a stable grasp with no slippage between the object and the fingers, the object will stay stationary with respect to the palm frame. Another advantage is the fact that the distance from the palm to the object is typically small, when compared to the distance to a fixed frame, *e.g.*, on the robot arm base. The choice of a reference frame far from the object would produce significant displacements with small rotation angles.

The next step of the algorithm aims to reduce the computation time of the algorithm by assuming that each finger is contacting the object inside a certain neighbourhood of its initial position, given by the vision system. This initial step thus consists of creating regions in the object on which each respective finger is predicted to lie, so that the algorithm only iterates over these regions instead of going through the whole object. This distance is set dynamically so that each finger is iterating over a minimum number of object points. The set over which finger number 1 will iterate is defined as $S^{(1)}$ and contains all the points $s_1^{(1)}, s_2^{(1)} \dots s_n^{(1)}$ whose distances to the finger contact locations $f^{(1)}$ are within the neighbourhood ε.

$$S^{(m)} = \{s_i^{(m)} \in O : \|s_i^{(m)} - f^{(m)}\| \leq \varepsilon\} \quad (2)$$

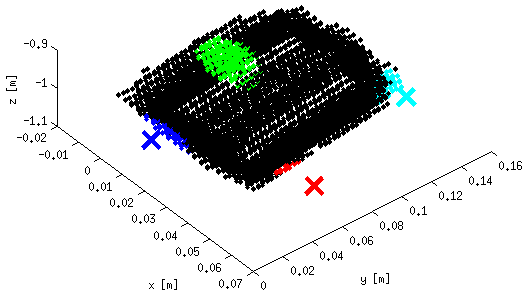


Fig. 2. Regions created in the object point cloud to minimise the computational effort

Equation (2) and Fig. 2 illustrate this first stage of the algorithm, where four fingers are contacting the object ($m = 4$) and the neighbourhood ε is set to be 15 mm. The point cloud O plotted in black is a standard 330 ml coke can and each contact location is shown as a coloured cross, with the mentioned neighbourhood regions $S^{(m)}$ in their respective colour. It can be seen, particularly on the red contact, that using the initial estimate of the object's location, the actual contact location does not sit on the object's surface. This error can be due to calibration errors or low accuracy of the vision system and it is what this method intends to correct.

Although the aim of the algorithm is to find the transform that fits the object point cloud into the contact locations given by the combination of the robot arm and hand's forward kinematics and the data obtained from the tactile system, it is computationally faster to transform the fingers to fit the object and then apply the inverse of the computed transformation to the object.

$$\begin{aligned} \mathbf{x} &= [\mathbf{q}, \mathbf{t}]^T \\ \mathbf{x} &= [q_w, q_x, q_y, q_z, t_x, t_y, t_z]^T \end{aligned} \quad (3)$$

To find the parameters $\mathbf{x}$ defined in (3) that describe the desired transform, the Levenberg-Marquardt iterative algorithm was used to minimise the objective function G. This transform consists, as usual, of a rotation and a translation, where the rotation is represented by a unit quaternion and the translation by a vector. The operation that describes a rotation $\mathbf{q}$ from point p to p' is shown in (4), where $\mathbf{q}^*$ is the conjugate quaternion [17].

$$p' = \mathbf{q}(p_x\mathbf{i} + p_y\mathbf{j} + p_z\mathbf{k})\mathbf{q}^* \quad (4)$$

The objective function G consists of the distances from the contact locations to its nearest point $s_i^{(m)}$ in its respective region $S^{(m)}$, as shown in (5).

$$G(\mathbf{x}) = \begin{cases} \min(\|(\mathbf{q}f^{(1)}\mathbf{q}^* + \mathbf{t}) - s_i^{(1)}\|) \\ \min(\|(\mathbf{q}f^{(2)}\mathbf{q}^* + \mathbf{t}) - s_i^{(2)}\|) \\ \dots \\ \min(\|(\mathbf{q}f^{(m)}\mathbf{q}^* + \mathbf{t}) - s_i^{(m)}\|) \end{cases} \quad (5)$$

Given that most handheld objects present some symmetry, fitting a small number of points to it often allows for more than one solution. The proposed method presents the advantage of finding a local minimum which is more likely to be a better solution than a global minimum that is too far off the initial pose acquired by the vision system to be deemed reasonable.

However, by using the LM method for this particular problem some caveats need to be noted: First, the minimum distance function G is not differentiable – we cannot know to which particular point in the region we are fitting the fingers because the closest point in the region is determined at each iteration, rendering the objective function to be discontinuous. To deal with this issue, the Finite Difference Jacobian[18] of the function has to be calculated using a forward difference approximation as shown in (6).

$$\mathbf{J} = (\nabla_h G)(\mathbf{x})_j = \frac{G(\mathbf{x} + h\|\mathbf{x}\|\mathbf{e}_j) - G(\mathbf{x})}{h\|\mathbf{x}\|} \quad (6)$$

Secondly, the approximate Hessian matrix $\mathbf{J}^T\mathbf{J}$ can, in some cases contain zeros in its main diagonal, rending $(\mathbf{J}^T\mathbf{J} + \lambda diag[\mathbf{J}^T\mathbf{J}])$ to become singular and thus not invertible. One such case arises when finding the partial derivative $\partial/\partial q_w$ of a *real* quaternion, *i.e.* quaternions $\mathbf{q}_1 = (1, 0, 0, 0)$ and $\mathbf{q}_2 = (1 + h, 0, 0, 0)$ represent the same rotation, originating a zero in the diagonal of $\mathbf{J}$. The alternative was to use the Moore-Penrose [19] pseudo-inverse $\mathbf{A}^+$, calculated using the Singular Value Decomposition as shown in (7), where Σ^+ is a diagonal matrix obtained by replacing each of the non-zero elements in the diagonal by its multiplicative inverse, leaving the zeros unchanged.

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \Longrightarrow \mathbf{A}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^* \quad (7)$$

The parameters $\mathbf{x}$ can be then calculated using modified Levenberg-Marquardt algorithm [18], [20] as follows:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - (\mathbf{J}^T\mathbf{J} + \lambda diag[\mathbf{J}^T\mathbf{J}])^+\mathbf{J}^T G(\mathbf{x}_i) \quad (8)$$

B. Computational Remarks

In order to enable the method to work in real-time, different strategies were employed to reduce the computational effort of the algorithm. The first simplification was based on the assumption that the first estimation from the vision system is near enough to allow the creation of regions on the object surface that are near to the finger tip contact location, as was described in (2). This reduction of the points to which each finger has to iterate to find the nearest point – in other words, calculate G – cuts down from some tens of thousands total points typically contained in a hand-held object, as given by the vision system, to regions containing around 150 points – a hundredfold improvement on the algorithm's performance.

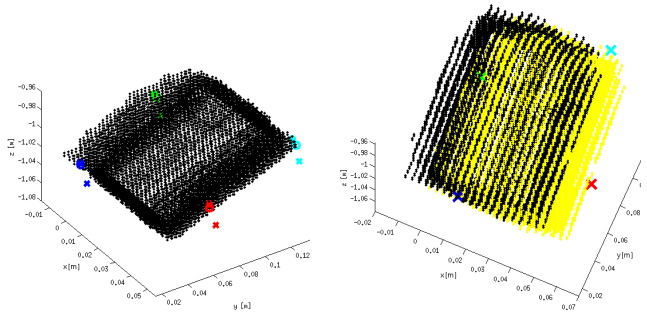
The second approach has also been described before and it consisted of finding the parameters that transform the fingers contact locations to fit the object, thus reducing significantly the amount of calculations required – as opposed to transform every point in the object's point cloud – and then execute the inverse transformation on the object. This approach reduced the needed calculations from the number of points in all regions, $\sum |S^{(m)}|$, to the number of fingers in contact with the object, m.

Thirdly, the use of quaternions instead of *roll-pitch-yaw (RPY)* angles or other representation of 3D orientation was chosen because, although it adds an extra variable to be determined when compared to *RPY* or *Euler-angles*, its advantages have been widely investigated and described in the literature [21]. In this particular algorithm the focus was on its benefits in terms of computational performance [22], [17]. As opposed to other representations, the calculations using quaternions involve no trigonometric functions or matrix conversions, requiring only one conjugation and two quaternion multiplication operations , as previously shown in (4). The possibility to easily invert and concatenate rotations using quaternions also weighted when choosing which rotation representation to use.

V. EXPERIMENTS AND RESULTS

A. Simulation

The above method was implemented both in simulation and with real data. In simulation, a standard coke can point cloud was used, and four points in its surface were taken and transformed to a known location, which served as input for the algorithm. To verify its accuracy, the initial points taken from the point cloud were plotted against the solution found by the algorithm. Fig. 3(a) shows the object point cloud in black, the chosen points in the surface as a filled dot, the transformed positions as a cross and the solution found by algorithm as a ring.



(a) Transformed finger locations (b) Transformed object

Fig. 3. Results using simulated data – Object point cloud in black. Original (filled dot) , displaced (cross) and corrected (ring) finger tip locations. Transformed object in yellow

It can be seen that the solution found corresponds very accurately to the original points. The choice of a coke can as the tested object was made on the grounds that a revolute object such as a cylinder would be the most prone to erroneous results. Fig. 3(b) shows the output of the method, where the object has been transformed using the inverse of the obtained transform to fit the contact locations on the finger tips. The black point cloud is again in its original position, as obtained by the vision system, and the yellow point cloud plots the same object after the transformation. After the transformation the object surface sits precisely on the computed finger locations.

The progress of the optimisation algorithm is plotted in Fig. 4. The algorithm requires 27 iterations to reach the desired error (distance from each finger to the object $d \leq 2.5$ mm) and converges to a maximum distance of 2.2 mm in finger 3 and an average 1.7 mm, starting from an initial estimate with a a maximum error of 12 mm in finger 1 and an average of 7 mm.

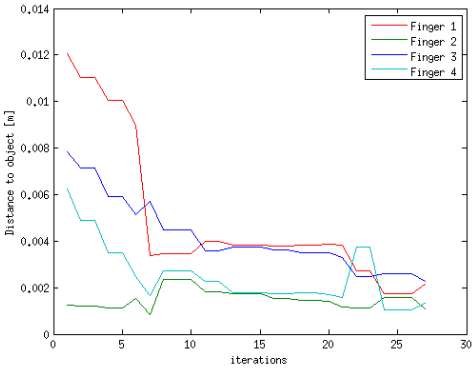


Fig. 4. Distance from each finger contact location to the object throughout the iterations using the Levenberg-Marquardt method

B. *Experiments using a real system*

Validation of the proposed method in a real system proved more challenging, given that it is hard to benchmark the results against a ground truth, as it would require a very accurate 6D tracking system. The method to verify the effectiveness of the method under a real environment was achieved by getting the robot to grasp a coke can and acquire a first estimate of its pose using a 3D camera. The scenario is a robotic platform consisting of the following (Fig. 5):

- A 4 Degrees of Freedom (DoF) Shadow robotic arm where the robotic hand is mounted on.
- A 24-DoF Shadow hand with ATI Nano17 6-axis force/torque sensors instrumented on the fingertips.
- A Microsoft Kinect camera is mounted on the left side of the robotic arm and is oriented to get a top view of the objects lying on a table.

A robot model for the shadow arm and hand, provided in the ROS platform [23], [24], was used which takes the values of the joint encoders and replicates the robot's configuration in a visualiser also available in the same platform. The object

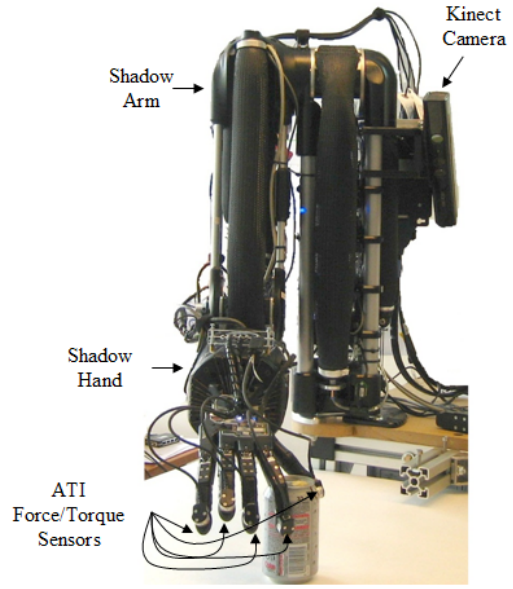


Fig. 5. Overview of the experimental setup of the multi-modal sensing system

point cloud was overlayed on the visualiser and, since the system was known to have a significant calibration error, the object model was clearly displaced from the finger tips. The algorithm was able to run in 10 iterations and converged to a pose that agreed with the observed situation: the transformed object's surface now coincided with the robot hand's contact locations. Fig. 6 shows the object in its initial pose in grey and the rectified pose in pink. The mean error in the initial estimate was 4.9 cm and the final was 5 mm. The chosen accuracy of the algorithm was thus a more conservative 5 mm when compared to the 2.5 mm used in the simulation tests described in the previous section to ensure convergence even with a deficient object model.

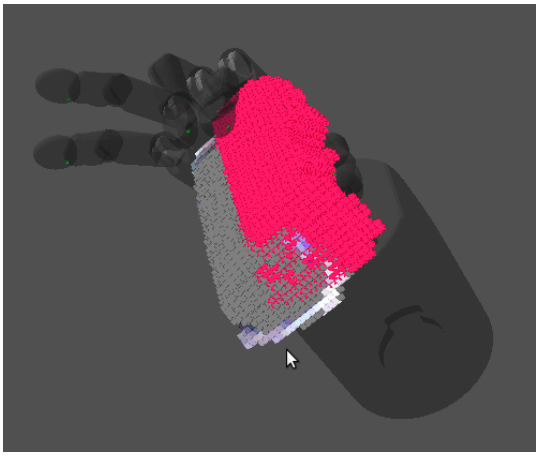


Fig. 6. Results using real data. Initial estimate in grey, solution in pink

The mean distance between the object and the fingers' contact locations was 4.9 cm in the initial estimate, being reduced to 5 mm after the correction of the object's pose. The accuracy of the algorithm has, of course, a trade-off

with its speed and depends on the accuracy of the first estimate obtained from vision. Regarding the computational performance, on a real system the average duration of the algorithm was 350 ms, with the step described in (2) average duration of 90 ms and the iterative algorithm averaging 260 ms.

VI. CONCLUSIONS AND FUTURE WORK

This paper presents a method to rectify the pose of a grasped object given the object model – acquired using a camera – and a number of contact locations – obtained using force/torque sensors which act like tactile sensors. This method aims to correct the errors arising from miscalibration, inaccuracy of the vision system or limitations such as occlusions. By amending the robotic system's knowledge of the object's pose, its accuracy and robustness when performing manipulation tasks will increase.

One of the limitations of this algorithm lies, as mentioned before, when dealing with symmetries on an object. Given that the object model may coincide with the contact locations in several positions, the objective function may have a vast number of solutions. The iterative method proposed will only find one that is near the initial estimation of the object pose which, in a system that provides a very coarse initial estimate, may not be the most correct. This limitation is closely related with the number of fingers contacting the object. As the number of contact locations increase, the possible object poses that comply with all of these locations becomes more limited, although never unique for certain types of symmetric objects.

Future work in this subject should firstly deal with the shortcomings discussed above. Other sources of information should be found that refine these possible transforms that fit the current contact locations. One such possibility is the testing of the contact normals – the surface normal of the object in a possible solution point should be parallel to the contact normal obtained from the finger tip sensor. Another feature to be added to this system could be the ability to detect if a transformed object collides with elements in the environment, *e.g.* fingers in the robot which are not touching the object. In addition, the inclusion of a second 3D camera could provide both a better object model and a more accurate first estimate of its pose.

Furthermore, in order to better validate this method, experiments to obtain statistical results involving rate of success of grasping and manipulation tasks when using the proposed algorithm should be carried out so that the advantages of this method are better established.

ACKNOWLEDGMENT

The authors of this paper would like to thank Prof. Véronique Perdereau and Dr. Guillaume Walck from Université Pierre et Marie Curie for their help in carrying out the tests on the Shadow arm and hand platform.

The research leading to these results has been funded by the HANDLE European project (FP7/2007-2013) under grant agreement ICT 231640 – http://www.handle-project.eu

REFERENCES

[1] R. C. Luo, Y. C. Chou, and O. Chen, "Multisensor Fusion and Integration: Algorithms, Applications, and Future Research Directions," in *Proc. of the 2007 International Conference on Mechatronics and Automation*. IEEE, Aug. 2007, pp. 1986–1991.

[2] P. K. Allen, A. T. Miller, P. Y. Oh, and B. S. Leibowitz, "Integration of Vision , Force and Tactile Sensing for Grasping," *International Journal of Intelligent Machines*, vol. 4, pp. 129–149, 1999.

[3] P. Hebert, N. Hudson, and J. Ma, "Fusion of stereo vision, force-torque, and joint sensors for estimation of in-hand object location," *IEEE Transactions on Robotics and Automation*, pp. 5935–5941, 2011.

[4] M. Prats, P. J. Sanz, and A. P. del Pobil, "Vision-tactile-force integration and robot physical interaction," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 3975–3980.

[5] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using SIFT features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345–352, Mar. 2009.

[6] P. K. Allen, A. Timcenko, and S. Member, "Automated Tracking and Grasping of a Moving Object with a Robotic Hand-Eye System," *IEEE Transactions on Robotics*, vol. 9, no. 2, 1993.

[7] A. Petrovskaya, O. Khatib, S. Thrun, and A. Ng, "Touch based perception for object manipulation," in *Robotics Science and Systems, Robot Manipulation Workshop*, 2007, pp. 2–7.

[8] K. Honda, T. Hasegawa, T. Kiriki, and T. Matsuoka, "Real-time Pose Estimation of an Object Manipulated by Multi-fingered Hand Using 3D Stereo Vision and Tactile Sensing," in *Proc. of the 1998 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, no. October, 1998, pp. 1814–1819.

[9] Microsoft, "Kinect for Xbox 360," 2010. [Online]. Available: http://www.xbox.com/en-US/kinect/

[10] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[11] Deliverable D23 of HANDLE European Project, "Visual and Tactile Perception System: Evaluation Report," Tech. Rep.

[12] W. E. L. Grimson and T. Lozano-Perez, "Localizing Overlapping Parts by Searching the Interpretation Tree," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, pp. 469–482, July 1987.

[13] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available:

[14] N. Burrus, M. Abderrahim, J. Garcia, and L. Moreno, "Object Reconstruction and Recognition leveraging an RGB-D camera," in *Proc. of MVA2011 IAPR Conference on Machine Vision Applications*, 2011.

[15] H. Liu, X. Song, J. Bimbo, and K. Althoefer, "Intelligent Fingertip Sensing for Contact Information Identification," in *Proc. of the Second ASME/IEEE International Conference on Reconfigurable Mechanisms and Robots (ReMAR 2012) (accepted paper)*, 2012.

[16] A. Bicchi, J. K. Salisbury, and D. L. Brock, "Contact Sensing from Force Measurements, Tech. Rep. 3, June 1993.

[17] J. Funda, R. Taylor, and R. Paul, "On homogeneous transforms, quaternions, and computational efficiency," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 3, pp. 382–388, June 1990.

[18] C. T. Kelley, "Iterative Methods for Linear and Nonlinear Equations," Jan. 1995.

[19] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*. Springer, 2003.

[20] K. Levenberg, "A method for the solution of certain problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, pp. 164 – 168, 1944.

[21] J. Schmidt and H. Niemann, "Using Quaternions for Parametrizing 3-D Rotations in Unconstrained Nonlinear Optimization," *VISION, MODELING, AND VISUALIZATION 2001*, pp. 399 – 406, 2001.

[22] E. Salamin, "Application of quaternions to computation with rotations," Stanford University Artificial Intelligence Laboratory, Tech. Rep., 1995.

[23] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," 2009.

[24] Willow Garage, "Robot Operating System." [Online]. Available: www.ros.org