Extracting grasping, contact points and objects motion from assembly demonstration

Damien Petit[1*] Ixchel G. Ramirez-Alpizar[1] Kensuke Harada[1,2]
Natsuki Yamanobe[2] Weiwei Wan[2] Kazuyuki Nagata[2]

***Abstract*— This paper presents a framework to extract the grasping, contact points and object parts motion from an assembly demonstration. With this framework the object parts are recognized and tracked using Augmented Reality (AR) markers. The data of the user's hand assembling the object are acquired with a motion capture device. The grasping and contact points are determined with the motion capture data, the models of the object parts and point cloud based algorithms. The functionality of the framework is demonstrated with an experiment where the user assembles two parts of a toy airplane. The grasping and contact points between the object parts are extracted and visualized. This framework aims at capturing the necessary data to reproduce the assembly motion on a dual-arm robot for future work.**

I. Introduction

Grasping and manipulating objects is considered to be an indispensable skill in many robotic systems. Over the past decades several algorithms have been developed to realize a stable robotic grasp. In order to achieve stability, different algorithms called grasping synthesis [1] have been established. The different approaches of these algorithms can be divided in two categories. The analytical approaches which are based on kinematic and dynamic formulations. And the empirical approaches which reproduce the human grasping motion and avoid the complex computation of the analytical algorithms. The framework presented in this paper collects the necessary data used in empirical approaches.

The key data required by empirical approaches are the fingertips positions of the human hand grasping the object and the grasping areas of the object. In this framework these data are captured using AR markers, a data acquisition glove (hereafter called "data-glove") [2], the 3D model of the object parts and point cloud based algorithms using the (Point Cloud Library) [3]. In the particular case of an assembly system, the objects manipulated may require certain configurations that generate extra constraints on the grasping problem. To correctly identify these extra constraints, the motion and contact points between the object parts being assembled are also captured. The capture of these contact points is one of the contributions of our work compared to other data extraction systems which focus only on the grasping and manipulation of one object [4].

Furthermore, most of the acquisition systems developed for human's movements are expensive due to the use of tactile gloves/sensors, motion capture systems, fine tracking cameras, etc. [5], and can not be used on different scenarios/applications. In contrast, we have developed a simple and versatile acquisition system that could be used in different situations and with a very low cost. The most expensive part of the framework is the data-glove which only costs 99 USD for 3 fingers tracking and 999 USD for upper body tracking [a]. This is much cheaper compared to other popular motion capture devices which cost around 10000 USD [2].

This paper is organized as follows. In section II the modules of the framework are presented. Next, in section III the data generated using the framework are shown while carrying out an assembly task. Finally, in section IV we discuss the results of the human demonstration and future work.

II. Framework and references

In this section we present the reference frames and the methods used in the framework to capture the motion, contact and grasping points of the object parts being assembled.

A. Framework

The framework is outlined in Figure 1. The framework uses as inputs an RGB camera, the models of the object parts and the data captured by the data-glove. The framework is used to capture the object and hand motion, the contact and grasping points as well as the fingertips position all along the assembly process. The framework operates as follows.

The RGB images are acquired by the camera and then processed through the `object recognition and tracking` module to recognize and determine the poses of the marked object parts and data-glove in the camera frame.

The poses of the object parts and the object parts models are used in the `contact points determination` module to determine the contact areas between the object parts.

The data-glove captures the fingertips position in its own frame reference. The `calibration` module calibrates the data-glove reference on the marker placed on the data-glove in order to obtain the fingertips position in the camera frame reference.

[1] Department of Systems Innovation, Graduate School of Engineering Science, Osaka University, 1-3 Machikaneyama-cho, Toyonaka, 560-8531, Japan.

[2] Manipulation Research Group, Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST), 1-1-1 Umezono, Tsukuba, 305-8560, Japan.

[*] Corresponding author e-mail: `damien.petit@hlab.sys.es.osaka-u.ac.jp`

[a] https://neuronmocap.com/products-fullwidth

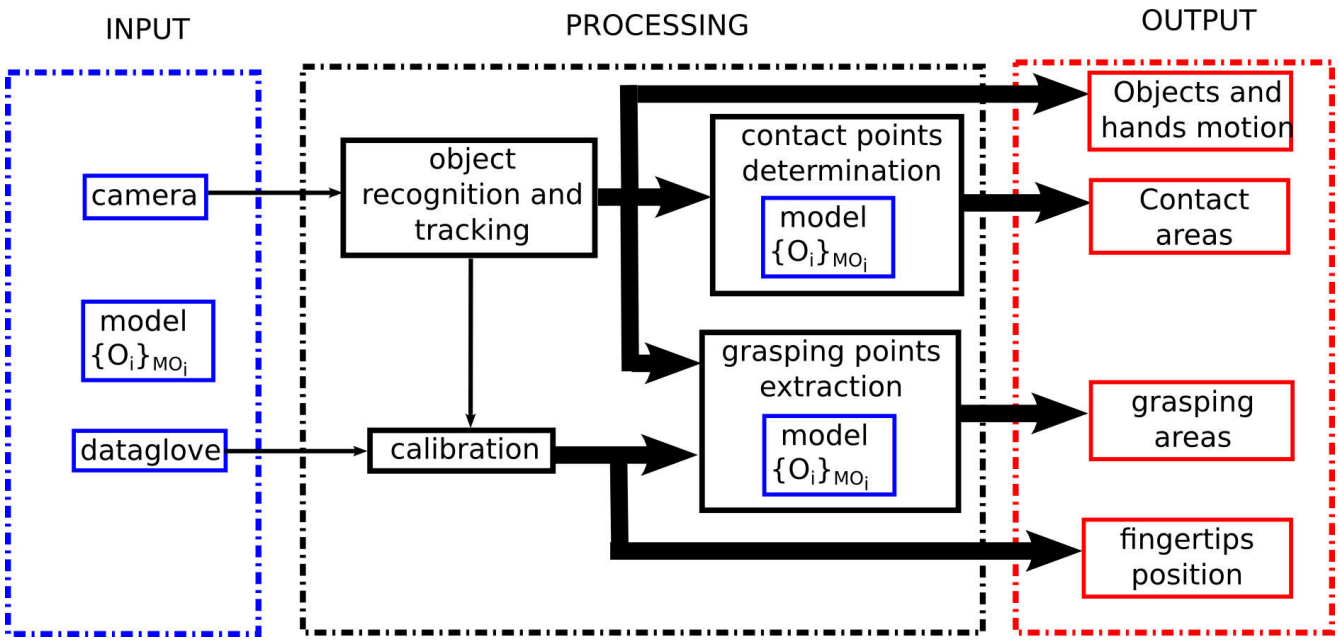

Fig. 1: Data flow of the framework

Once the fingertips and object parts poses are known in the camera frame the grasping areas are determined using the object parts models and a nearest neighbor search in the `grasping points extraction` module.

All the modules of the framework are integrated with the Robot Operating System [b] ROS [6]. In the following sections the reference frames and modules used in the framework are described in more details.

B. Frame definitions

The reference frames used in this work are shown in Figure 2. We consider the RGB camera frame `C`, the AR markers' frame attached to the object parts assembled by the user `M1` and `M2` and the marker's frame attached to the data-glove `MG`. The frames captured by the data-glove `F1`, `F2`, `F3` represent respectively the thumb, index and middle fingertips of the user. The frame `G` represents the data-glove internal frame.

C. Object recognition and tracking

To identify and localize the marked object parts in the RGB images, we use the ArUco library [c]. ArUco relies on printed black and white square fiducial markers (as seen in Figure 2), instead of natural textures or key points [7], to provide a robust and accurate pose estimate. The algorithm consists in detecting the square contour, finding the optimal threshold in the bimodal image, extracting the binary code, and comparing it to the known dictionary, to infer the marker identity (in our case either $M1$, $M2$ or MG). Finally, the marker pose ${}^{C}\mathbf{T}_{M}$ (with $M = \{M1, M2, MG\}$) is estimated by iteratively minimizing the image plane projection error of the four square corners, with the Levenberg-Marquardt method.

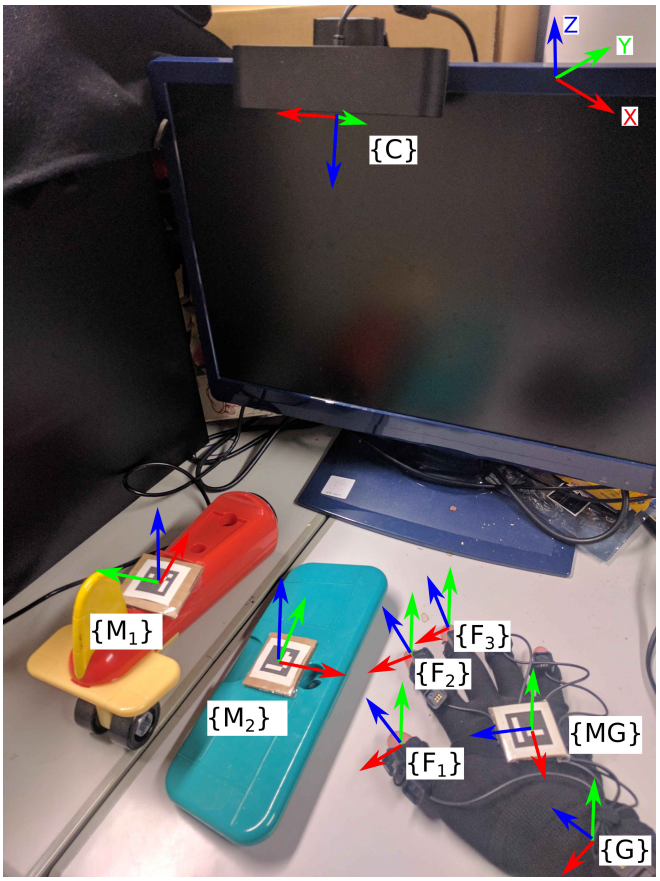

Fig. 2: Reference frames used in this work.

[b] http://www.ros.org/
[c] http://www.uco.es/investiga/grupos/ava/node/26

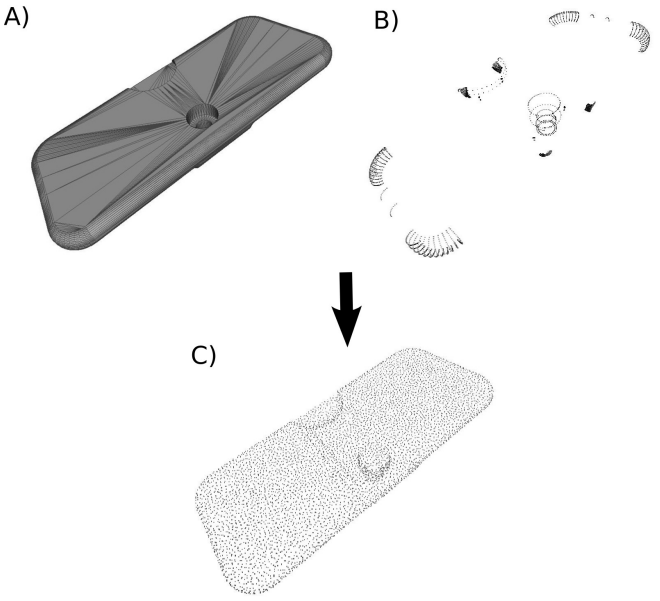

Fig. 3: Top wing point cloud generation. A) Mesh model. B) Vertices of the mesh model C) Point cloud model based on the mesh model after sampling

D. Point cloud model generation

In this framework the object parts models are represented using point clouds instead of meshes. With the rise of the RGB-D cameras and LIDAR [d], point cloud data acquisition is becoming more and more common and affordable. Even though triangulation technique exists [8] to generate a mesh from a point cloud, the process remains complex and time consuming, which makes it difficult to use in real-time application. This is especially true for large scene coming from slam algorithm [9].

On the other hand, most of the object parts are given in a polygon/mesh representation. This is particularly true for assembly parts made or used in factory. In this paper, the object parts models of the toy airplane are originally given in such a representation. In order to obtain a point cloud representation of these parts, we create a new layer populated with a point sampling of the mesh model based on [10]. The result is shown in Figure 3 for the top wing part.

E. Data-glove and calibration

To capture the fingertips positions we use the data-glove *Perception Neuron* system [e]. The data-glove is equipped with 8 sensors, 2 on the thumb and index, 1 on the remaining fingers and 1 on the back of the hand. In this study we are interested with the fingertips data position of the thumb `F1`, index `F2` and middle finger `F3`.

As mentioned in section II-A the data-glove captures the hand motion data in its own frame `G`. To use the captured data in our framework we calibrate the internal reference frame `G` of the data-glove on the marker placed on it `MG` thus determining ${}^{MG}\mathbf{T}_G$. The determination of this transformation is done empirically with the visual feedback of the fingertips frames `F1`, `F2`, `F3`.

[d]http://spectrum.ieee.org/automaton/robotics/robotics-hardware/sweep-lidar-for-robots-and-drones

[e]https://neuronmocap.com/

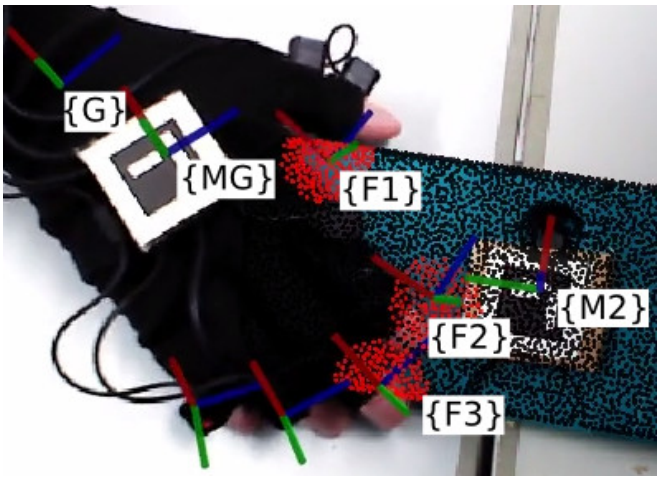

Fig. 4: The top wing part model (in black dots) overlays the real top wing. The red dot represents the grasping points of the top wing centered on the fingertips frames `F1`, `F2` and `F3` captured by the data-glove.

F. Grasping points determination

The grasping points of the object parts are determined using an algorithm based on point cloud and the fingertips positions captured by the data-glove after calibration.

First, the model of the object part (explained in section II-D) is overlayed on the real object part by using the pose of the AR marker in the camera frame ${}^{C}\mathbf{T}_M$. Then, the point cloud model of the object part is organised in a kd-tree to realize a nearest neighbor search based on [11] centered on each fingertips positions $F1$, $F2$ and $F3$ inside a sphere of radius rg.

The nearest neighbor search is realized in the the object part frame `M`. This allows the point cloud to be organised in a kd-tree only one time since the point cloud model does not change in its own frame due to the rigid nature of the object part. The point cloud is organised in a kd-tree during the initialization of the framework thus decreasing the computation time of the nearest neibor search at runtime.

The results of the grasping points determination are shown in Figure 4.

G. Contact points determination

The contact points detection is realized using the point cloud model of the object parts and their poses in the camera frame ${}^{C}\mathbf{T}_{M1}$, ${}^{C}\mathbf{T}_{M2}$, respectively for the body and top wing parts. The contact detection is also based on a nearest neighbor search.

First, both of the models are overlayed on the real object parts using ${}^{C}\mathbf{T}_{M1}$ and ${}^{C}\mathbf{T}_{M2}$. Then, one of the model part is organised in a kd-tree in order to realize an optimized nearest neighbor search centered on the points of the other

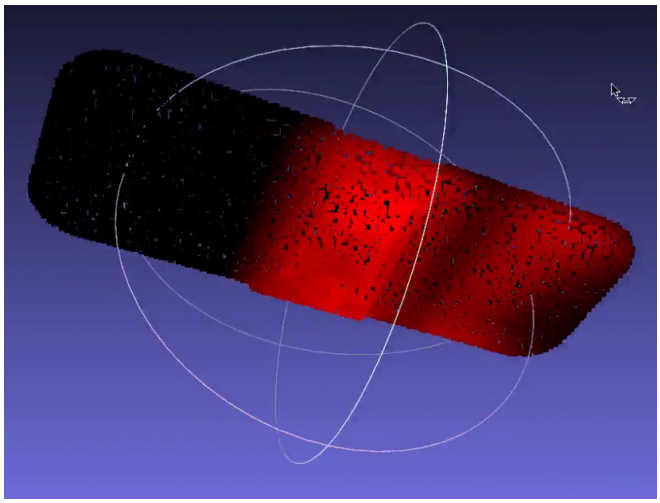

Fig. 5: Top wing part model contact with slide assembly motion. In this case we observe that the edge of the wing was in contact with the body part and that the assembly task was realized by sliding the top wing from the side of the body part.

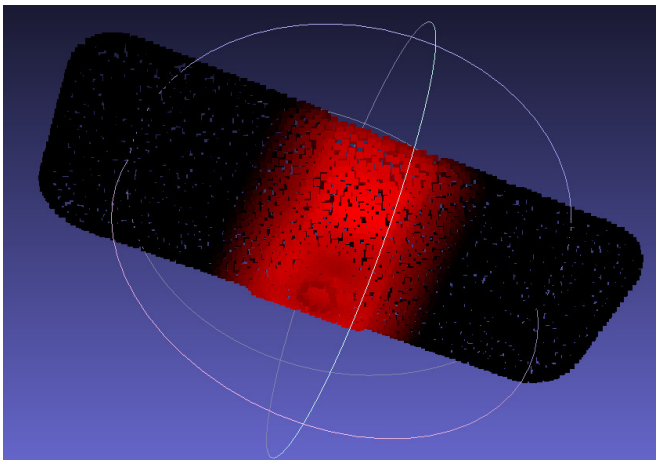

Fig. 6: Top wing part model contact without slide assembly motion. In this case the edge of the plane is not coloured meaning that there was no contact at this location. The assembly task was performed by placing the top wing from the top of the body part, no sliding motion was realized.

model part. The search is realized within a sphere of a radius rc.

Like explained in section II-F the kd-tree is organised only one time which allows a fast collision detection at runtime.

H. Contact and grasping points visualisation

In this section, the generated visual models based on the results of the grasping points and contact points are explained.

To observe and quickly analyse the obtained results the framework is equipped with a visualisation tool. The tool generates a colored model of the object parts with the contact and grasping points based on the assembly task realised by the user. The results can be observed in Figures 8, 9 and 10.

1) Grasping points model: To generate the grasping points model, we define a visualisation value v for each points of the model. In each frame v is incremented by 1 for each point where a contact is detected. When the assembly task is over v is equalized from 0 to 255 in order to visualize the results in a colored channel (in our case the red channel) of the point cloud model like seen in Figure 10. With this method, an area colored with a strong intensity corresponds to a long grasp.

2) Contact points model: The contact points model is generated on the same principle as the grasping points model. In this case, v is incremented by 1 to each contact points detected between the object parts models at every frame. Like with the grasping points model v is equalized at the end of the assembly task. The results can be observed in Figures 5, 6, 8 and 9.

III. Experiment

In order to demonstrate the functionality of the framework a user was asked to assemble the top wing and body parts of a toy airplane placed on a table (as seen in Figure 2). For the experiment we use the data-glove which is connected to a computer running Windows 10 64-bit. The data is sent via a network cable through a ROS topic to another computer equipped with Ubuntu 14.04 64-bit where the framework is running. The computer where the framework is installed has 8G memory and a i5-4460 CPU. We use a Orbbec Astra S camera [f] to acquire the rgb images with a resolution and frequency of $640 * 480$ @ $30Hz$. The camera is equipped with a depth sensor but the depth information is not used in this framework. The subject was a male aged of 31 with prior knowledge in robotics.

Figure 7 shows the top wing of the toy airplane being assembled to the plane's body by the user. The grasping and contact points are determined with the algorithms described respectively in section II-F and II-G.

The model generated for the grasping points is shown in Figure 10. The contact points model are shown in Figures 8 and 9.

IV. Discussion

The experiment shows that the framework realizes its function by providing the grasping and contact points of the object parts as well as their motion while being assembled. One of the advantages compared to other data acquisition system [4] is its simplicity to set-up and its affordability. Indeed, most of the other data acquisition system are based on expensive materials to track the user's hand, fingertips and the grasping points of the object parts. Our system is based on printed AR markers which are easy to make and

[f] https://orbbec3d.com/

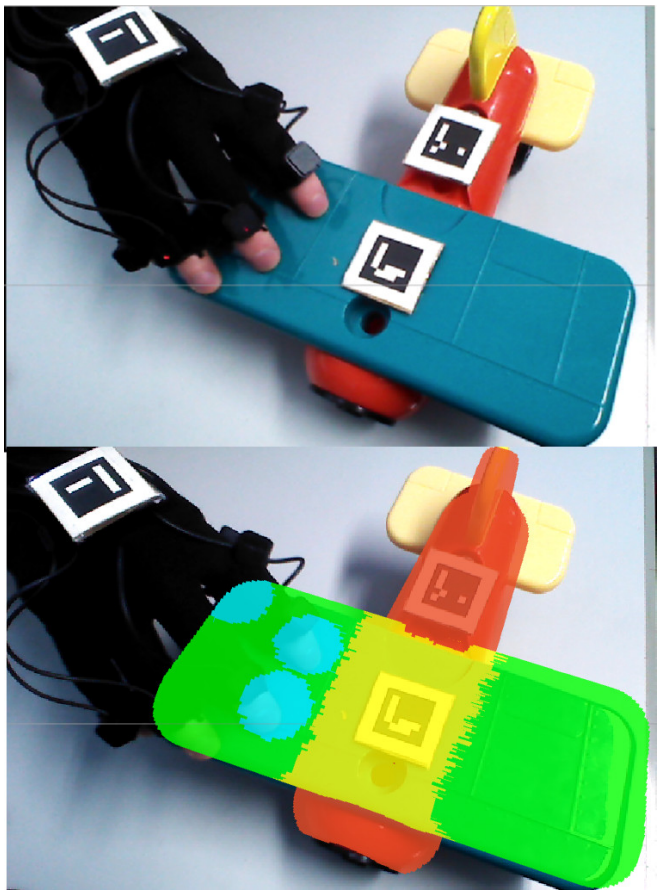

Fig. 7: Grasping and collision points during assembly demonstration. Blue areas correspond to the grasping points. Yellow area corresponds to the collision points. The orange body and green top wing models are overlayed on their real respective parts

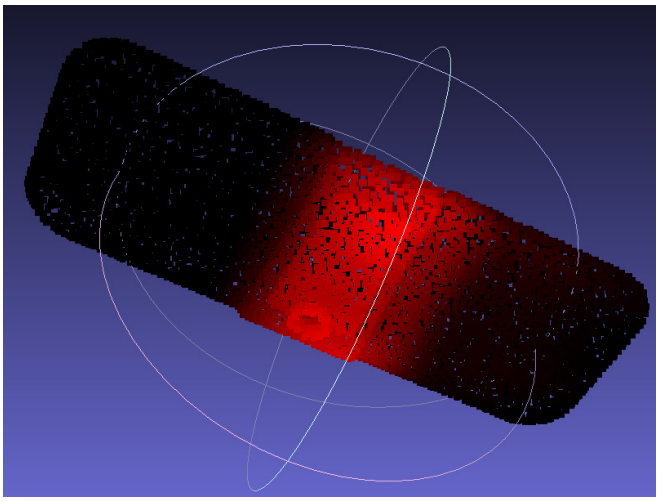

Fig. 8: Top wing part contact points results. The intensity of the model reflects the duration of the contact. In this case we observe that the edge of the wing was in short contact with the body part of the plane during the assembly task.

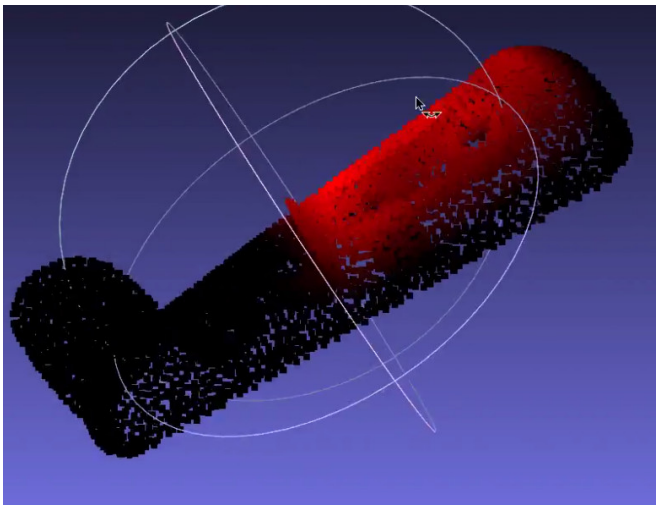

Fig. 9: Plane body part contact points results. The red area indicates the contact with the top wing part of the plane. We can notice that the red color is more intense on the edge of the fixture part.

place on the object parts and data-glove. The user's fingertips are tracked with a data-glove which is cheap and mobile compared to other systems. Our framework also detects the contact points between the object parts being assembled without the need to place expensive sensors on the object parts.

The visualisation tool implemented in the framework offers useful information from a grasping point of view but also from an assembly point of view.

The information obtained from the grasping areas and the fingertips motion will be used to synthesis a stable grasp [12]. But they also offer an insight on the user's behaviour while realizing an assembly task compared to realizing a grasping task by studying the difference in grasping areas location. The grasping model offers also the possibility to study whether or not the user's fingertips slip during an assembly, grasp task or both.

The contact points model allows to quickly understand how the assembly was realized. For example in Figure 5 we can see that the assembly was made by sliding the top wing part on the body part hence the slight red areas (corresponding to a slight contact) at the top wing's edge. Whereas in Figure 6 we observe that the assembly was made with no sliding contact between the body part and the top wing part.

Several aspect of the framework could be improved. Indeed, one advantage of using a data-glove over a vision system to capture the fingertips positions while assembling the object parts is to avoid the problem of occlusion of the fingers. But, in order to capture the fingertips and object parts positions in a common frame, the data-glove must be calibrated on the camera frame. In this study the calibration of the data-glove is realized using a AR marker. Model fitting techniques would improve the calibration of the data-glove which would lead to an improvement of the grasping points accuracy.

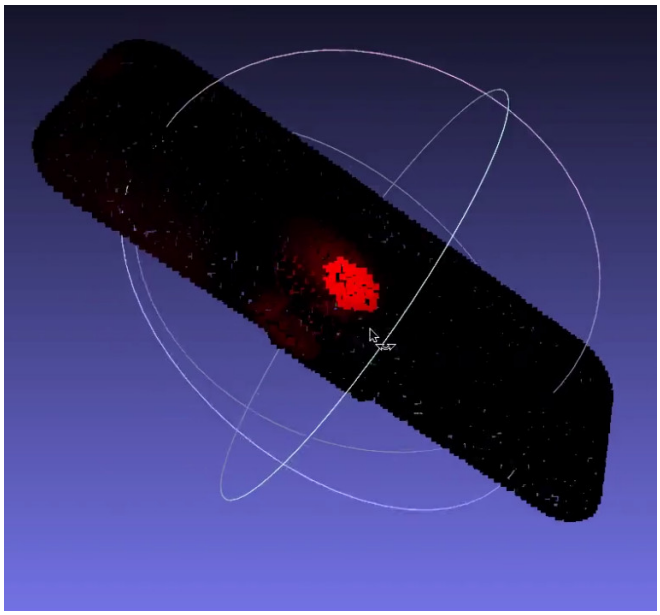

Fig. 10: Top wing part grasping points results. We can observe the intense red area corresponding to one of the user's finger grasping the top wing part.

Moreover, in the framework the AR markers provide a reliable tracking results but they also restrain the motion and the grasping areas of the user. Markerless tracking methods [13] [14] would allow the user to grasp and assemble the object parts more naturally.

The contact and grasping model could also be improved by taking into account distance from the fingertips to the contact surface as a weight in the incrementation of the v value.

V. Conclusion

This paper shows the functionality of the proposed framework to capture the grasping and contact points as well as the motion between two object parts being assembled by a user. The framework offers a quick understanding of the assembly realized by generating a contact and grasping model of the object parts. Future research will focus on improving the framework by implementing a markerless object tracking algorithm and using the framework to capture and replicate the assembly motion on a Dual-Arm robot.

Acknowledgment

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

[1] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3D object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, pp. 326–336, 2012.

[2] L. Dipietro, A. M. Sabatini, and P. Dario, "A survey of glove-based systems and their applications," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 38, no. 4, pp. 461–482, 2008.

[3] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE Int. Conf. on Robotics and Automation ICRA*, pp. 1–4, 2011.

[4] D. R. Faria, R. Martins, J. Lobo, and J. Dias, "Extracting data from human manipulation of objects towards improving autonomous robotic grasping," *Robotics and Autonomous Systems*, vol. 60, pp. 396–410, mar 2012.

[5] M. Ehrenmann, R. Zöllner, O. Rogalla, S. Vacek, and R. Dillmann, "Programming service tasks in household environments by human demonstration," in *11th IEEE International Workshop on Robot and Human Interactive Communication (ROMAN)*, pp. 25–27, 2002.

[6] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, vol. 3, 2009.

[7] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, pp. 2280–2292, jun 2014.

[8] F. Remondino, "From point cloud to surface: the modeling and visualization problem," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XXXIV, pp. 24–28, 2003.

[9] M. Meilland and A. I. Comport, "On unifying key-frame and voxel-based dense visual SLAM at large scales," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3677–3683, 2013.

[10] M. Corsini, P. Cignoni, and R. Scopigno, "Efficient and flexible sampling with blue noise properties of triangular meshes," *IEEE Transactions on Visualization and Computer Graphics*, 2012.

[11] M. Muja and D. G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," in *International Conference on Computer Vision Theory and Application*, pp. 331–340, 2009.

[12] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-Driven Grasp Synthesis -A Survey," *Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2014.

[13] J. Issac, M. Wüthrich, C. G. Cifuentes, J. Bohg, S. Trimpe, and S. Schaal, "Depth-Based Object Tracking Using a Robust Gaussian Filter," *IEEE International Conference on Robotics and Automation (ICRA) Stockholm*, 2016.

[14] T. Mörwald, J. Prankl, A. Richtsfeld, M. Zillich, and M. Vincze, "BLORT - The Blocks World Robotic Vision Toolbox," in *Proc. ICRA Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation*, 2010.