Flexible Robot Sealant Dispensing Cell using RGB-D sensor and off-line programming

Perla Maiolino[a], Richard Woolley[b], David Branson[c], Panorios Benardos[c], Atanas Popov[c], Svetan Ratchev[c]

a. Computing Department, Goldsmiths University of London SE14 6NW, UK
b.Centre for Aerospace Manufacturing, University of Nottingham, Nottingham NG7 2RD, UK
c. Department of Mechanical, Materials and Manufacturing Engineering, University of Nottingham, NG7 2RD, UK

Abstract

In aerospace manufacture the accurate and robust application of sealant is an integral and challenging part of the manufacturing process that is still performed by human operator. Automation of this process is difficult and not cost effective due to the high variability in the parts to operate and also the difficulty associated with programming industrial robotic systems. This work tries to overcome these problems by presenting an AOLP (Automatic Off-Line Programming) system for sealant dispensing through the integration of the ABB's proprietary OLP (Off-Line Programming) system RobotStudio with a relatively new RBG-D sensor technology based on structured light and the development of a RobotStudio add-on. The integration of the vision system in the generation of the robot program overcomes the current problems related to AOLP systems that rely on a known model of the work environment. This enables the ability to dynamically adapt the model according to sensor data, thus coping with environmental and parts variability during operation. Furthermore it exploits the advantages of an OLP system simplifying the robot programming allowing for faster automation of the process.

keywords: AOLP, RGB-D sensor, Sealant dispensing

1. Introduction

In aerospace manufacture the accurate and robust application of sealant is an integral part of the manufacturing process. Sealant must be applied to numerous components of the aircraft including areas of wet assembly, such as the wings (fuel tanks), joints, fasteners and interfaces to prevent ingress, corrosion and reduce fatigue [1].

Applying sealant is a challenging operation for assembly automation due to the requirement of dispensing a continuous and uniform bead along an often complex geometric path to which there is limited access, requiring the execution of a number of ergonomically complex operations. Operations of this nature, particularly in legacy products, rely heavily on the skill or rather craftsmanship of a human operator. Certainly some of these processes can be carried out by a suitably designed robotic assembly cell. Indeed predefined and highly repetitive tasks and processes are commonplace for robotic automation in high volume manufacturing. Operations such as pick and place, welding, painting and dispensing are considered basic tasks.

In these types of applications robots are usually programmed using a teach pendant which, although it is an intuitive programming approach and doesn't require high programming skills, it does suffer from the reliance on the human programmer and furthermore to make changes to the robot path, once made, is laborious and time consuming. Thus this type of automation is not cost effective for aircraft manufacturing, which encloses complex small volume processes where the geometry of the parts and the required operations vary considerably. Moreover, the fixtures and feeders, that guarantee a fixed position of the workpiece for the robot operations, are expensive. Hence, their removal or reduction from the manufacturing process in favor of a higher level of flexibility in the automation system would present an overall capital gain [2].

In this perspective, good sensing abilities are ubiquitous to provide robots with an increased level of autonomy. For example vision systems are highly suited for providing Robot's environment information as well as for the recognition of objects and their features providing information about their position and pose, allowing it to more reliably act in an unconstrained cell. Furthermore, to ensure any automated process is carried out effectively after manual operation, vision can be used for in-line structural quality inspection to confirm that the assembly's sub-components are mounted within design specification [3].

Machine vision has been used widely in manufacturing and production, in a number of applications [4]. Indeed the state of the art presents various solutions that use different techniques (both 2D than 3D) such as monocular vision [5], CCD cameras [6] or laser range imaging [7]. Many types of vision systems and range sensors are available today, but the design and integration of them needs to take considerations of many issues such as accuracy, speed, reliability, and cost as well as environment operating constraints such as lighting, temperature and humidity, requiring to find a reasonable trade off according to the requirements of the particular applications.

Beside good sensing abilities it is necessary to develop systems that can reduce the time and difficulty associated with programming industrial robotic systems allowing robotic automation to be justified even in low volume production manufacturing scenarios where frequent reprogramming is necessary. In this respect Offline programming software (OLP) [8], that gives the possibility to model and represent graphically the robot and other equipment, generate programs and, then simulate the robot behavior for a given task, have been developed in recent years both as commercially available product (for example KUKA Sim for Kuka, RobotStudio for ABB and MotoSim for Motoman Delmia from Dassault Systems, RobCAD from

Technomatix Technologies and Robotmaster from Jabez Technologies) as well as in research field.

In particular in [14], a system for CAD-based OLP is presented where robot programs are automatically generated from a graphical description of the robot paths over a 3-D CAD model of a given robotic cell. The paper explores the issues involved in robot motion representation and its automated extraction from a given CAD drawing, data mapping between the CAD model and the real environment as well as the ability to automatically generate the necessary robot paths and programs.

Hand-guided robot operations, workpiece CAD model and triangulation sensor for detection of workpiece contour features are combined for robot programming in deburring applications [15].

The OLP system presented in [16] uses the geometry computing functions of CATIA and the simulation function (robot kinematics, collision detection, etc.) of KUKA Sim Pro and it is focused on robotic drilling applications in aerospace manufacturing. In particular, the special requirements for robotic drilling in aerospace manufacturing are analyzed to improve the position accuracy and to take in account actual shape and position of the part, the hole position is corrected according to measurement data of reference holes by using bilinear interpolation models.

In [17] the development of OLP programs and the evaluation of several OLP software packages such as, RobotStudio from ABB Robotics, Delmia from Dassault Systems and a Matlab based OLP system and RinasWeld from Kranendonk Production System is presented, demonstrating the effectiveness of automated OLP techniques for the robotic welding process.

Unfortunately OLP systems are still in their infancy undergoing significant amount of research and development, and employing OLP system usually requires great programming effort and some capital investment that can significantly reduce the feasibility in using them with low volume production.

A novel idea that is gaining ground and which could improve OLP methods is related to using computer automation for performing all of the manual tasks, termed Automated OLP (AOLP) [9-11]. AOLP allows one to automate many of the manual functions required by conventional OLP, but currently it strongly relies on accurate modeling of the work environment. To overcome this constraint, additional sensors could be deployed to and around the robot to map the work environment, enabling the robot to be more context aware and provide boundaries for developing self-programming algorithms [11].

In [8] an AOLP solution to automatically plan and program a robotic welding system with high DOF manipulators is presented. It uses a CAD model as input, and is able to generate the complete robotic welding code without any further programming effort. AOLP can generate collision-free and singularity-free trajectories for the complete system including the linear rail, auxiliary positioning robot, and welding robot automatically.

All of the above systems can be applied in situations where the robot's surrounding environment are known a priori and well modeled. Furthermore, sometimes human help is necessary to regenerate the robot program whenever a change in the original scenario occurs.

In this paper the problem of the robotic automation of the sealant dispensing in aircraft manufacturing is taken into account. As stated before, this procedure is still performed by human operator due to the high variability in the wing parts (in term of shape and dimensions) and also to the difficulty associated with programming industrial robotic systems. The work presented is a step toward an AOLP system for sealing dispensing

through the integration of the ABB's proprietary OLP system RobotStudio with a relatively new RBG-D sensor technology based on structured light [7].

In particular the development of a new add-on for RoboStudio has been designed to integrate the results coming from the implementation of an object recognition and pose estimation algorithm, based on Point Cloud Library [22], for the generation of the robot path. From one side this allows to overcome the problem, related to AOLP systems, to rely on a known model of the work environment giving the opportunity to dynamically adapt the model according to the sensors data. This further enables the systems to cope with environmental and part variability in an automated way. If further exploits the advantages of an OLP system by simplifying the robot programming allowing for the robotic automation of the process.

The selection of a low cost RGB-D sensor relies on the potential for a data rich manufacturing environment provided by a high deployment density of these low cost sensors as a key enabling feature of the future cyber physical systems that would be part of an Industry 4.0 SMART factory [12]. Indeed it is foreseen that advances in computer vision capabilities will be accelerated with the use of low cost RGB-D sensors and the open implementation of visual algorithms [13].

The paper is organized as follows: Section 2 gives an overview of the system architecture, the RGB-D sensor and Robotstudio add-on, Section 3 shows the recognition and pose estimation algorithm and Section 4 is about the Robotstudio add-on implementation and the integration with vision system. Section 5 presents the results, conclusions follow.

2. **The Flexible sealant dispensing cell: system overview**

The automation of the sealant dispensing has been thought as a partially automated process where the human operator presents the bracket to be operated to the robot, after having performed on it other manual operation. Each part is provided by an RFID tag that is scanned before to present the part to the robot and that allow to upload from a database the corresponding part information, in particular its CAD model and where, with respect to the CAD model, the sealant must be dispensed. This information will be used for the dynamic generation of the robot program by the AOLP system presented in section 4.

Figure 1 shows the designed sealant dispensing cell and the physical setup used for the experiments of the developed system. In particular an ABB IRB 6640 robot has been selected for sealant dispensing (please note that the sealant dispensing nozzle is not shown in the figure because it has been specifically designed for the considered task and it is covered by an non-disclosure agreement prior to patent application). The RGB-D camera has been placed over the shelf in order to be able to perform the recognition and pose estimation of a generic trailing edge rib bracket placed on the shelf in a random position. Figure 2 presents all the steps related to the sealant dispensing automation system that will be explained in details in the following paragraphs.

a) b)

Figure 1: a) The designed sealant dispensing cell. b) The experimental setup of the dispensing cell.

2.1 RGB-D sensor

A low cost RGB-D sensor (ASUS Xtion PRO LIVE) was employed for the vision system. RGB-D sensors provide color information as well as the estimated depth for each pixel. The sensor technology is based on structured light [18] (originally developed by PrimeSense). The sensor projects an IR-light pattern on the scene, and the light returns distorted, according to the position of the objects in the scene. Various algorithms are used to triangulate and extract 3D information from the CMOS image sensor. In [30] it is possible to find the technical specifications of the Asus Xtion Pro Live, for a comparison of sensor types the reader is directed to [19,20], where the authors note that the future development and exploitation of these types of sensors lies in the development of advanced and robust algorithms rather than the refinement and cost increase of the sensors themselves.

The sensor was placed above the working shelf and relative to the robot arm (ABB IRB6640) as shown in Figure 1. This position ensures a good number of detection points describing the bracket. As reported in [21] the depth image resolution of the sensor is inversely proportional with the distance between the object and the sensor and the point cloud density will decrease with increasing distance of the objects' surface from the sensor (it is inversely proportional to squared distance from sensor).

The recognition and pose estimation algorithm has been developed using the Point Cloud Library (or PCL). PCL is an open source project [22] for 2D/3D image and point cloud processing released under the terms of the 3-clause BSD license and is open source software. PCL is a cross-platform library that can be deployed on a number of different operating systems (Linux, MacOS, Windows, Android and iOS). In addition, the architecture allows for integration into a number of controllers and smart systems.

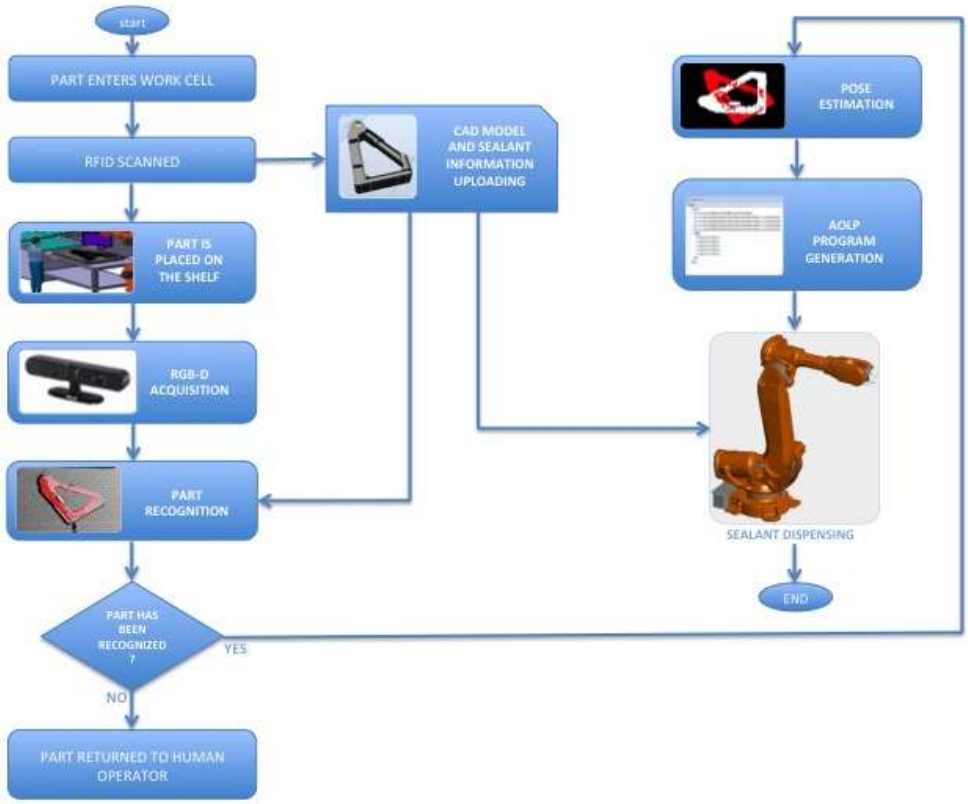

Figure 2: The flow chart of the sealant dispensing automation system

To integrate machine vision system with the OLP software Robotstudio, the RGB-D sensor has been connected to a standard Windows PC through USB interface. The PC receives the acquired point cloud from the sensor, it performs the necessary calculations for recognition and pose estimation of the part and sends the transformation matrix corresponding to the actual pose of the bracket to Robotstudio developed add-on through the network.

2.2 ABB RobotStudio and add-on for off-line programming

ABB RobotStudio is an off-line robot programming and simulation tool for ABB robots. Through RobotStudio it is possible to program the robot using a graphical environment and simulate the developed program on a virtual controller to check issues related to reaching, accessibility and collision at the programmed targets points and along the path. Furthermore, when the code is fully developed off-line it can then be downloaded to the robot controller for execution.

RobotStudio supports Microsoft's Visual Studio for Applications. This gives the possibility to change and enhance the behaviour of RobotStudio, by developing add-ons for specific purposes related to user

requirements. Add-ons are developed in C# or VB.NET and built for obtaining a "dll" that improves the capabilities of RobotStudio. The developed add-on program for the flexible sealant dispensing cell is responsible for getting the bracket pose information from vision system, automatically generating the robot trajectories for performing sealant dispensing on it and asking for the execution of the operation.

3. Recognition and pose estimation

Several algorithms for 3D object recognition have been proposed in literature [31]. In particular they can be divided according to the fact they use local and global approach. In the local approach distinctive keypoints are extracted from the 3D surface of the model and the scene. Each keypoint is associated with a 3D descriptor related to the keypoint local neighborhood. The scene and model keypoints are matched together based on the descriptors for obtaining correspondences that are clustered together on the base of geometrical constraints under the hypothesis of model-to-scene rigid transformation. The clustered correspondences define a model hypothesis. In global methods a single descriptor for the whole object shape is computed and for this reason a pre processing of the scene must be performed for extracting individual object instances. For both the approaches an object hypotheses verification phase take place for rejecting false detections. In particular one hypothesis at time is considered and it receive a consensus score depending on the amount of scene points corresponding to the transformed model points.

Between all the existing approaches, for our recognition and pose estimation, we selected the algorithm presented in [23]. The choice has been driven by the fact that this algorithm instead of considering each model hypothesis individually, take into account the whole set of hypotheses as a global scene model maximizing the number of the recognizing models and, then, outperforming all the state of the art algorithms. The complete pipeline of the algorithm implemented for our purposes is presented in Figure 3, where all the steps performed by the algorithm are reported which will be explained in details in the following. This pipeline is based on local features, although it must be pointed out that the proposed algorithm can be straightforwardly plugged into pipelines based on global features or recognition pipelines combining several descriptors with different strengths as long as a 6DOF pose is provided.

Given the scene point cloud acquired by the RGB-D sensor (Figure 6a), recognition of the part in the scene is performed to check if the part is present on the shelf and if the RFID tag corresponds to the right part.

Fig.3. The recognition and pose estimation algorithm pipeline.

In particular after the acquisition of the scene point cloud a plane segmentation filter is performed to eliminate all the points belonging to the shelf and to consider only the point belonging to the bracket. This is important for both removing noise and reducing the number of cloud points, and thus computational time required for the more complex following keypoints selection phase.

The keypoints selection phase takes place both for the model and filtered scene point cloud. The objective of this phase is to decide which point of the cloud will be selected for the computation of the descriptors. The

keypoints are extracted at uniformly sampled positions at a sampling distance of 0.03 mm.

The next step is to compute the desired local feature, i.e., descriptor, for each of the keypoints (descriptor computation phase in Figure 3). There are several local descriptors available to choose from and their effectiveness depends on the scene and the object to recognize. For our application the SHOT descriptors were used [24] due to its computational efficiency, descriptive power and robustness with respect to other type of descriptors. The SHOT local descriptors have been computed at each keypoint over a support size specified by radius 0.02 mm.

After the computation of the local descriptors for both the keypoints belonging to the model and the scene, a matching phase is executed to find correspondences between them (descriptors matching phase in Figure 3). This is completed using a nearest neighbor search between descriptors performed by calculating the Euclidean distance between them [25]. This method allows handling of multiple instances within the same model on the scene.

Given a list of correspondences a clustering stage is performed using the correspondence grouping (CG) algorithm [26]. The algorithm iteratively groups subsets of correspondences based on checking the geometric consistency of pairs of correspondences: starting from a seed correspondence $c_i = \{p_{i,M}, p_{i,S}\}$, where $p_{i,M}$ and $p_{i,S}$ are respectively, the model and scene 3D keypoints for c_i, and looking to all the correspondences not yet grouped, the correspondence $c_j = \{p_{j,M}, p_{j,S}\}$ is added to the group, for which c_i is the seed, if the following relation is verified:

$$\left| \left\| p_{i,M} - p_{j,M} \right\|_2 - \left\| p_{i,S} - p_{i,M} \right\|_2 \right| < \varepsilon \tag{1}$$

where, ε is a threshold that represents the inlier tolerance for the consensus set. Each subset of correspondences obtained through CG defines a model hypothesis that will be part of the hypothesis set (H) evaluated in the following hypothesis verification (HV) stage. To optimize the final cardinality of H and to improve the computational efficiency of HV, an additional threshold parameter τ in the CG algorithm is used to discard subsets supported by too few correspondences. For each cluster CG calculates the transformation matrix identifying the 6DOF pose estimation of the model in the scene. Iterative closest point (ICP) [27] algorithm is used for refining the 6DOF obtained. At this point H is ready for the HV phase [23] that for each hypothesis belonging to H decides if it is verified or rejected (i.e., false positive).

4. RobotStudio add-on for AOLP programming

The RobotStudio add-on has been developed using C# and the Controller API provided by RobotStudio. The pipeline for RobotStudio add-on and its integration with machine vision system through UDP socket communication is shown in Fig. 5.

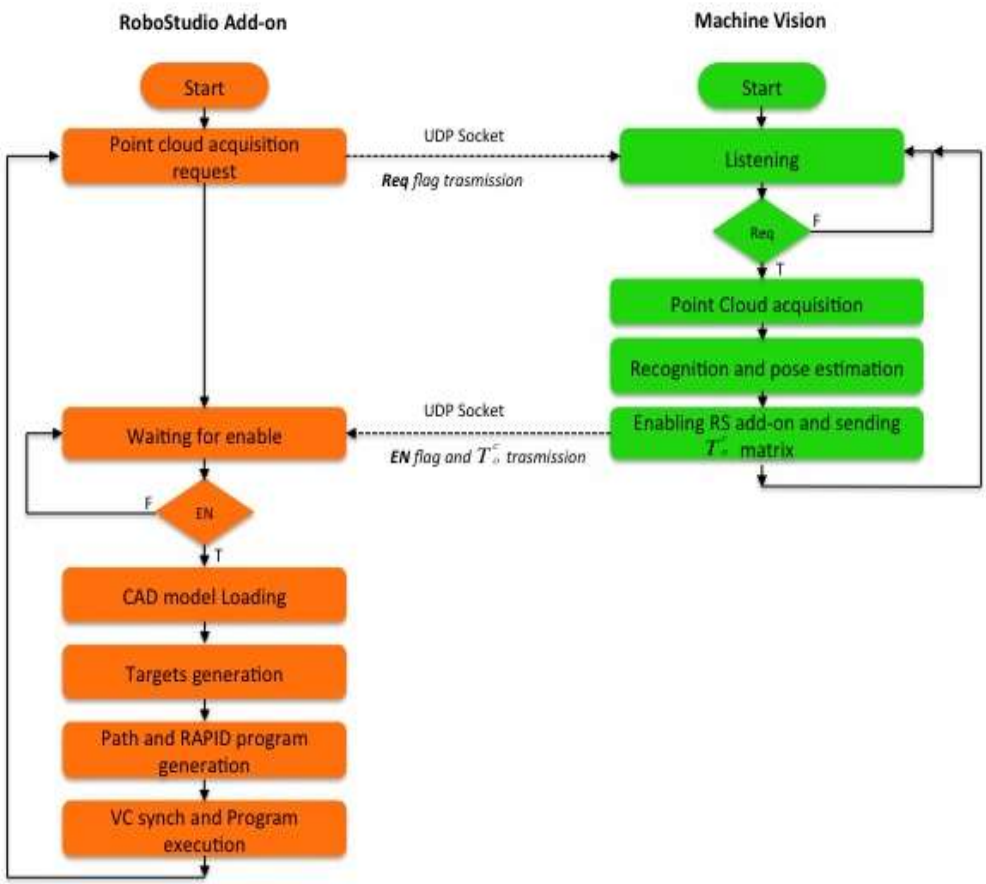

Fig. 4. The software integration of RobotStudio add-on and Machine Vision software for recognition and pose estimation

The beginning the add-on sends an *acquisition request* to the machine vision system, to receive information related to the object to operate and its actual pose. This is important for loading the right CAD model (as discussed before, in the presented scenario the sealant must be dispensed on more than 50 different types of brackets) in the same pose of the recognized object. After the request, the add-on remains in a *listening* state. The machine vision system after performing recognition and pose estimation sends back an *enable command* and the information related to the actual pose of the object that allow the add-on to perform the subsequent calculations (explained in details in the following subsections) and the sealant dispensing robot program generation.

4.1 Steps of OLP: CAD model loading

From the recognition and pose estimation algorithm the add-on receives the information of the matrix T_O^C corresponding to the transformation between object and camera frame. From this the homogenous transformation matrix, T, is in form:

$$T = \begin{bmatrix} \mathrm{R} & \mathrm{t} \\ 0 & 1 \end{bmatrix} \tag{2}$$

Given the T_O^C, it is necessary to calculate T_O^W corresponding to the transformation matrix between the object (i.e., bracket) and the world frame. The relationship between transformation matrices is expressed by Eq. 3 and shown in Fig. 4.

$$T_O^W = T_O^C T_C^W \tag{3}$$

where T_C^W is the transformation matrix between the camera frame and the world frame. T_C^W has been determined using a 6 DoF laser tracker for measuring the accurate position of the camera with respect to the world coordinate frame. In particular a target device integrating a reflector together with a set of LEDs on its surface have been fastened on RGB-D camera before, and after on the robot base (corresponding to our world reference frame); the 3 position parameters (x, y, z) and the 3 orientation parameters (pitch, yaw and roll) for both RGB-D camera and world reference frame with respect to laser tracker coordinate frame, have been determined through a CMOS digital camera system operating infrared (IR) radiation, mounted on the laser tracker and able to continuously follow the target device and capture the images of the IR LEDs mounted on it performing the absolute measurement of the parameters. On the base of the calculated parameters is possible to obtain the two transformation matrix T_L^W and T_L^C from which it is possible calculate T_C^W in an analogous way as in Eq. 3.

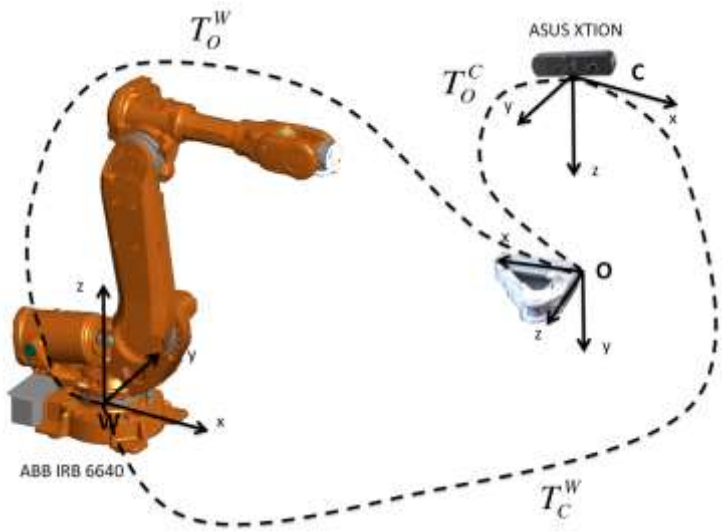

Fig. 5 Camera position with respect to the robot. W represents the world frame, C represents the camera frame and O the object frame. T corresponds to the transformation matrix between frames and are indicated by superscript and subscript.

On the basis of the above transformations, the add-on automatically loads the bracket CAD model in the actual pose in RobotStudio simulation environment.

4.2 Steps of OLP: Targets generation

In the proposed scenario, we consider to know in advance where it is necessary to dispense the sealant. For this reason, with each bracket a CAD model, which defines the points that constitute the position corresponding to sealant dispensing site, is enclosed in the database

The add-on reads the points related to the recognized bracket from the database and creates corresponding robot target points in the simulation environment of RobotStudio. As the points are stored with respect to the CAD model frame, the add-on applies to them the same homogeneous transformation applied to the CAD model for loading it in the actual object position.

Since a target can have any solution, in terms of joint configurations, it is necessary first of all to determine if the target is reachable (i.e., if exists at least one configuration solution) and if so, to select the best one relating to the specific task. Indeed when using OLP for programming complex trajectories it is necessary to take into account that 6-DOF industrial robots are subjected to several constraints coming from their kinematic structure (i.e., joint limits, singularities, workspace limits/reachability) [28]. To successfully accomplish the robot task the planned trajectories must be free from the above kinematic constraints. To overcome the above limitations the possible solutions were limited to only those that guarantee that the tool keeps the same orientation during the entire path, although the development of a specific path planning algorithm is out of scope of this work. For each target the joint configuration solutions, provided by RobotStudio, are determined considering the TCP frame aligned with the target frame. For this reason an additional relative transformation of the target frame is performed to align the TCP frame in the right position (i.e., in the opposite direction of the object surface normal calculated in the target point).

4.3 Steps of OLP: Path and RAPID code generation

To generate the path for the robot, RobotStudio API gives the possibility to create *Move* instructions by using the created targets. These robot movements are programmed as pose-to-pose movements, i.e. “move from the current position to a new position”. The path between these two positions is then automatically calculated by RobotStudio. Choosing the appropriate motion type instruction, i.e., linear, joint or circular, specifies the basic motion characteristics, such as the type of path. The remaining motion characteristics are specified by defining the arguments of the instruction and in particular; position data (robot target corresponding to the end position), speed data (desired speed), zone data (position accuracy), tool data (e.g. the position of the TCP) and work-object data (e.g. the current coordinate system). After the path creation phase all the RobotStudio objects and instructions are translated to RAPID language and a RAPID code is generated automatically.

4.4 Steps of OLP: VC Synchronization and Program execution

After the above phases, the synchronization of the RAPID program on the virtual controller is performed to simulate the robot motions. The simulation is an important part of the process, since it allows to verify the program without the use of the physical robot. The execution of the program from the physical robot requires a calibration procedure for correcting the position and orientation errors coming from differences between the actual kinematic model of the robot and the nominal one used in the simulation environment [29]. Please note that, since the scope of this work was the integration of machine vision and RobotStudio, a calibration procedure to assess the real kinematic model of the robot has not been performed and the add-on has been tested only with the virtual controller.

5. Results

The flexible robot programming system was tested to evaluate its capabilities. All the experiments have been performed in the real scenario unless the final robot motions execution that has been evaluated in the simulation environment provided by RobotStudio. The reason of this is that RobotStudio generates the robot motions based on the virtual kinematic model that is stored in the simulation environment. This kinematic model should be corrected with a proper absolute calibration procedure for compensating the differences with respect to the real robot kinematic model. Without this necessary calibration, when the RobotStudio generated code is downloaded in the real robot controller, it can lead to accuracy errors in the robot motions. As stated before, the absolute calibration procedure has not been performed, but it will be considered as future activity.
The bracket rib was placed randomly on the shelf and the RobotStudio add-on started. Figure 6a shows the bracket point cloud acquired by the RGB-D sensor.

a) b) c)

Fig. 6: a) Show the acquired scene point cloud: the bracket is placed on the shelf in a general orientation. The points belonging to the shelf will be removed through the plane segmentation filtering phase. b) The resulting point cloud after recognition phase. Highlighted in red is the recognized occurrence of the bracket in the scene presented in a). c) The actual position of the bracket, in red, with respect to the bracket model used for recognition phase, in white, is shown.

After plane segmentation filtering only the points belonging to the bracket are considered and the vision algorithm performs the recognition of the bracket in the scene (Figure 6b). The model of the bracket, used for the recognition, has been acquired with the RGB-D sensor and stored in a DB that contains the model of all the possible brackets to operate. Figure 6c shows the actual position of the bracket w.r.t the model used for recognition stage.

After recognition and pose estimation phase the RS add-on selects the CAD model corresponding to the recognized bracket and loads it in the right position according to the actual pose determined by the vision system algorithm (Figure 7a). The target points corresponding to the place where dispensing sealant on the bracket (in this case around the bore) are read from a file in the database, transformed as per the CAD model of the bracket and created in RS (Figure 7b). Finally, in Figure 7c the path for robot motion is generated and the corresponding RAPID code created. Finally the synchronization with the virtual robot controller is performed and the motion executed (Figure 7d). The motion has been simulated at a velocity of 100mm/s. This velocity has been selected based on previous experiments with the designed dispensing tool.

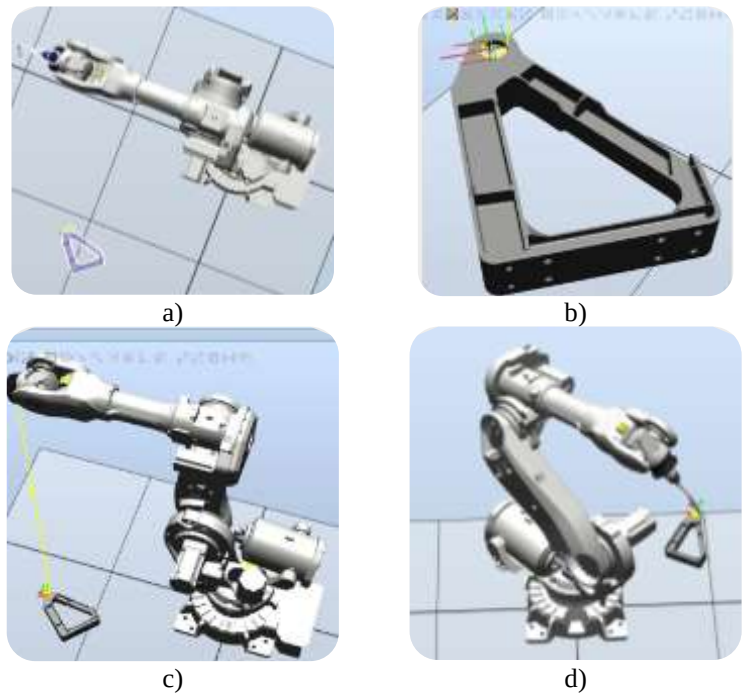

Fig. 7. a) CAD model loading. b) Targets setting. c) Path generation d) Motions execution

It is important to highlight that the position accuracy of the object is affected by the error of the RGB-D sensor and pose estimation algorithms (Figure 6b). The accuracy of the camera has been assessed in the operating conditions. The camera has been placed at 0.836 m from the shelf where the object is usually placed in order to have a reasonable field of view and in the same time to avoid hindrance to robot movements. The light conditions in the workshops were partially controllable because of a skylight in the sealing. It has been decide to leave the available neon lights always turned on.

5 camera acquisitions of the shelf were made at the shortest possible timespan one after the other at four different times during the day (10 AM, 1 PM, 2 PM, 4:30 PM) in order to test the camera with different levels of light and different light directions.

Table 1 shows for each condition the mean, the standard deviation, the median and the error between the real distance of the shelf from the camera frame origin and the median. The minimum error is 5 mm in the morning and in the late afternoon while the maximum error is 9 mm at 1 PM.

Table 1: Results of the calculations of camera accuracy under each light conditions, values in mm.

	AVG	STD	MEDIAN	ERROR
MORNING LIGHT (10 AM)	0.8315	0.0073	0.8310	0.005
LUNCH LIGHT (1 PM)	0.8264	0.0040	0.8270	0.009
EARLY AFTERNOON LIGHT (2 PM)	0.8257	0.0055	0.8290	0.007
LATE AFTERNOON LIGHT (4:30 PM)	0.8320	0.0082	0.8310	0.005

To achieve greater accuracy a proper light system and additional sensors should be used for online refinement and correction of the model during dispensing such as force/torque sensors or laser scanners. In this work a top-down approach in which the RGB-D sensor provides the OLP system a model with the environment results in a reasonable accuracy. Improving accuracy is out of the scope of the presented work, which is focused on the integration of OLP system and vision sensor, but is the object of future work plans.

6. Conclusions

We have presented an AOLP system for automated sealant dispensing that, through the development of an add-on for RobotStudio, integrates a low cost RGB-D sensor to dynamically generate robot control programs for use in variable part systems.

The developed Robotstudio add-on allows overcoming the problem for existing OLP systems that rely on accurate models of the environment, thanks to the integration of a RGB-D vision sensor in the loop allowing for a more dynamic program generation. The possibility to dynamically generate the robot program without a-priori precise model of the environment enables robotic automation in the process of sealant dispensing in aircraft manufacturing where the high variability in the parts to operate limits its use leading to a low production rate that thanks so robotic automation could be increased.

The main advantage of the developed system is first of all that it does not require having a-priori knowledge of the environment and the part to operate because it dynamically generates the robot program according to the recognized object and its current pose, helping to reduce the cost due to reprogramming. The developed system, therefore, enables robotic automation in aircraft manufacture where the high variability of the part to operate doesn't allow using the common "automotive" approach. The recognition and pose estimation algorithm has been developed on the basis of the algorithm proposed in [23]. This algorithm have been selected because, with respect to other proposed algorithm for object recognition and pose estimation it uses a global hypothesis verification that allow it to be more reliable and increase the recognition performance even for strongly occluded objects. The system has been tested in the simulation environment, because in order to have a real assessment of the performances with the real robot, an absolute calibration procedure has to be done for adapting the robot virtual kinematic model in RobotStudio with the robot real kinematic model. The absolute calibration and the performances assessment will be performed as future work as well as the use of a laser scanner for on-line correction of the path during sealant dispensing.

Acknowledgements

We gratefully acknowledge funding from Innovate UK No. 38279- 263183 (VIEWS). The authors would also like to thank Mr. Kevin Walker and Mr. Geoffrey Bexon for their contribution in the establishment of the robot station.

References

1. Lucas F. M. da Silva, Andreas Öchsner, Robert D. Adams (2011). Handbook of Adhesion Technology, Springer Science & Business Media, 10 Jun 2011.

2. Talavage, Joseph. Flexible Manufacturing systems in practice: design: analysis and simulation. Vol. 26. CRC Press, 1987.

3. Maiolino, P., Woolley, R. A., Popov, A., & Ratchev, S. (2015). Structural Quality Inspection Based on a RGB-D Sensor: Supporting Manual-to-Automated Assembly Operations. SAE International Journal of Materials and Manufacturing, 9(2015-01-2499).

4. Malamas, E. N., Petrakis, E. G. M., Zervakis, M., Petit, L. & Legat, J. D. "A survey on industrial vision systems, applications and tools." Image Vis. Comput. 21, 171–188 (2003).

5. Lei, L. "A machine vision system for inspecting bearing- diameter." Fifth World Congr. Intell. Control Autom. (IEEE Cat. No.04EX788) 5, 3904-3906 (2004).

6. Wong, a. K. C., Rong, L. & Liang, X. "Robotic vision: 3D object recognition and pose determination." Proceedings. 1998 IEEE/RSJ Int. Conf. Intell. Robot. Syst. Innov. Theory, Pract. Appl. (Cat. No.98CH36190) 2, 1202–1209 (1998).

7. R., Ceres, R. & Pons, J. L. "A vision system based on a laser range- finder applied to robotic fruit harvesting." Mach. Vis. Appl. 11, 321–329 (2000).

8. Pan, Z., Polden, J., Larkin, N., Van Duin, S., & Norrish, J. (2012). Recent progress on programming methods for industrial robots. Robotics and Computer-Integrated Manufacturing, 28(2), 87-94.

9. Ames, Arlo L., Elaine M. Hinman-Sweeney, and John M. Sizemore. "Automated generation of weld path trajectories." Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, 2005. (ISATP 2005). The 6th IEEE International Symposium on. IEEE, 2005.

10. Polden, J., Pan, Z., Larkin, N., Van Duin, S., & Norrish, J. (2011). Offline programming for a complex welding system using DELMIA automation. In Robotic Welding, Intelligence and Automation (pp. 341-349). Springer Berlin Heidelberg.

11. Larkin, N., Pan, Z., van Duin, S., & Norrish, J. (2013, July). 3D mapping using a ToF camera for self programming an industrial robot. In Advanced Intelligent Mechatronics (AIM), 2013 IEEE/ASME International Conference on (pp. 494-499). IEEE.

12. Stork, A. "Visual Computing Challenges of Advanced Manufacturing and Industry 4.0". Comput. Graph. Appl. IEEE 35, 21–25 (2015).

13. Shellshear, E. "Maximizing Smart Factory Systems by Incrementally Updating Point Clouds." Comput. Graph. Appl. IEEE (2015).

14. Neto, P., & Mendes, N. (2013). Direct off-line robot programming via a common CAD package. Robotics and Autonomous Systems, 61(8), 896-910.

15. Dietz, T., Schneider, U., Barho, M., Oberer-Treitz, S., Drust, M., Hollmann, R., & Hägele, M. (2012, May). Programming System for Efficient Use of Industrial Robots for Deburring in SME Environments. In Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on (pp. 1-6). VDE.

16. Zhu, W., Qu, W., Cao, L., Yang, D., & Ke, Y. (2013). An off-line programming system for robotic drilling in aerospace manufacturing. The International Journal of Advanced Manufacturing Technology, 68(9-12), 2535-2545.

17. Larkin, N., Milojevic, A., Pan, Z., Polden, J., & Norrish, J. (2011). Offline programming for short batch robotic welding.

18. Geng, J. (2011). Structured-light 3D surface imaging: a tutorial. Advances in Optics and Photonics, 3(2), 128-160.

19. Gonzalez-Jorge, H., Riveiro, B., Vazquez-Fernandez, E., Martínez-Sánchez, J., & Arias, P. (2013). Metrological evaluation of Microsoft kinect and asus xtion sensors. Measurement, 46(6), 1800-1806.

20. Daniilidis, K., Maragos, P. & Paragios, N. “Computer Vision - ECCV 2010.” (Springer, 2010).

21. Khoshelham, K. & Elberink, S. O. “Accuracy and resolution of kinect depth data for indoor mapping applications.” Sensors 12, 1437–1454 (2012).

22. Rusu, R. B. & Cousins, S. “3D is here: point cloud library.”(2011).

23. Aldoma, Aitor, et al. "A global hypotheses verification method for 3D object recognition." *Computer Vision–ECCV 2012*. Springer Berlin Heidelberg, 2012. 511-524.

24. Tombari, F., Salti, S., & Di Stefano, L. (2010, September). Unique signatures of histograms for local surface description. In European conference on computer vision (pp. 356-369). Springer Berlin Heidelberg.

25. Muja, Marius, and David G. Lowe. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration." VISAPP (1) 2 (2009).

26. H. Chen and B. Bhanu: “3D free-form object recognition in range images using local surface patches”, Pattern Recognition Letters, vol. 28, no. 10, pp. 1252-1262, 2007.

27. Zhang, Zhengyou (1994). "Iterative point matching for registration of free-form curves and surfaces". International Journal of Computer Vision (Springer) 13(12): 119–152. doi:10.1007/BF01427149.

28. Kržič, P., Pušavec, F., & Kopač, J. (2013). Kinematic Constraints And Offline Programming In Robotic Machining Applications. Tehnicki vjesnik/Technical Gazette, 20(1).

29. Pan, Z., Polden, J., Larkin, N., van Duin, S., & Norrish, J. (2011). Automated offline programming for robotic welding system with high degree of freedoms. In Advances in Computer, Communication, Control and Automation (pp. 685-692). Springer Berlin Heidelberg.

30. https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/specifications/

31. Y. Guo, M. Bennamoun, F. Sohel, M. Lu and J. Wan, "3D Object Recognition in Cluttered Scenes with Local Surface Features: A Survey," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 11, pp. 2270-2287, Nov. 1 2014.