International Journal of Innovative
Computing, Information and Control
Volume **16**, Number **5**, October **2020**

ICIC International ©2020 ISSN 1349-4198
pp. 1769–1785

PERCEPTIVE AUGMENTED REALITY-BASED INTERFACE FOR ROBOT TASK PLANNING AND VISUALIZATION

Ahmed Eslam Solyman[1], Khaled Mohamed Ibrahem[2]
Mostafa Rostom Atia[2], Hassan Ibrahim Saleh[1] and Magdy Raouf Roman[3]

[1]Radiation Engineering Department
Egyptian Atomic Energy Authority
Cairo, P.O. 11787, Egypt
ahmedesolyman@gmail.com

[2]Department of Mechanical Engineering
Arab Academy for Science Technology and Maritime Transport
Cairo, PO box 2033, Egypt
mrostom1@aast.edu

[3]Department of Mechanical Power Engineering
Faculty of Engineering
Helwan University
Cairo, P.C. 11718, Egypt
magdy_roman@m-eng.helwan.edu.eg

Received April 2020; revised August 2020

Abstract. *Robots are often used to replace humans in hazardous, uncomfortable and tedious works. However, due to the low repeatability and high versatility in High-Mix Low-Volume (HMLV) manufacturing, the cost of robots' programming is still significant. Programming methods based on getting in direct contact with robots, although intuitive, are not always allowed for safety concerns. Moreover, unstructured environments pose an additional challenge where a collision-free path including, not only the end-effector but also the entire robot links, is required. In this research, an intuitive semi-automatic offline robot programming method based on Augmented Reality (AR) and Stereo Vision (SV) system is proposed. The method has a simple user interface where the operator can intuitively program a remotely located robot in unstructured environments. The human-robot interface system is formulated by information fusion of stereo vision data, proposed AR algorithm, and human intuitiveness. Experiments are conducted to evaluate the proposed system. Candidate operators are given a task of programming a remotely located 6-DOF robot to accomplish a free-of-collision pick-and-place operation in unstructured environment. The results showed effectiveness in robotic task implementation.*
Keywords: Robot programming, Human-Robot Interaction (HRI), Augmented reality, Collision-free path, Stereo vision

1. **Introduction.** Robots are used to handle a wide range of tedious, dangerous, and time-consuming tasks instead of humans [1-3]. Nowadays, robots become cheaper, and their technology has been developed to a great extent; however, (re)programming is still such a difficult, time-consuming process. This makes it unfavourable for many industrial aspects especially Small and Medium Enterprises (SMEs), which have little autonomous capabilities. This includes short production cycles, small lot sizes, unstructured environments and lack of experts with robotics knowledge. On the other hand, there are circumstances in which the operators are not allowed to exist in the robot environment for safety

DOI: 10.24507/ijicic.16.05.1769

reasons. The task of finding an intuitive user-friendly interface for Human-Robot Interaction (HRI) that is able to suit different working environments for non-skilled operators is not simple [4-6].

In industrial robot application, there are two main categories of conventional robot programming methods, namely, offline programming and online programming [7]. Offline programming is a programming method where operators do not interfere with the robot working environment and the program is written on an external PC. Traditionally, CAD model of the workpiece and Virtual Reality (VR) are used in offline programming where the planned task is simulated before execution. This method is, however, inflexible to unstructured environments due to the need to re-model, calibrate and fine-tune physical entities to compensate for the discrepancies [8-12]. Online robot programming, on the other side, is where the robot program is created or updated while the robot is online. The two famous conventional online methods are the walk-through and lead-through programming. In walk-through programming, the operator is in direct physical contact with the robot where joints of the powered robot with disengaged brakes are manually moved to the desired positions. This method is not suitable for hazardous working environment. In addition, a few robots can offer zero-torque control [13-16]. In lead-through programming, robots are moved through teach pendant (e.g., a joystick) where movements are recorded and played back. Although the process allows the operator to be at distance from the working environment, it is considered unintuitive, cumbersome and time-consuming. Moreover, both lead-through and walk-through teachings usually involve trial-and-error that highly affects the accuracy [7,17-20].

Robot programming methods can be seen from different views according to the degree of autonomy in generating the programming code. This can be classified into manual, automatic, and semi-automatic programming methods. In manual programming, the user creates the robot program manually which is typically performed without the robot and then loaded into it afterward. Manual programming is usually time consuming, unintuitive, and needs experienced operators with robot programming background [21]. In automatic programming, the user has little or no direct control over the generated code. Automatic programming usually needs sophisticated software and hardware, as well as developed infrastructure. This includes environment sensing, robot control, accuracy handling, exceptions processing, etc. [22]. Semi-automatic programming, on the other hand, is when the operators assist the robots to solve unexpected problems [23]. Human robot interaction techniques in such methods play a vital role.

In order to overcome the drawbacks and limitations of the conventional robot programming methods the current research presents an intuitive AR-assisted human-robot interaction to facilitate programming remotely located serial robots in unstructured environments. The research has potential importance in vital applications such as dealing with nuclear waste and radioisotope production which were carried out conventionally using master and slave manipulators. Such highly complex tasks necessitate the intervening of the operators and cannot be programmed fully automatically or manually. The proposed semi-automatic system is developed to reduce cognitive fatigue and to facilitate working at distance in such applications. Moreover, it is designed to help operators with little programming experience to accomplish fast and intuitive robot programming and allow them to focus only on tasks definition. This is also of potential importance in small and medium enterprises.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes the system architecture overview. Section 4 discusses the use of methods and materials. Section 5 discusses experimental work and results. Finally, Section 6 concludes the paper and proposes future research opportunities.

2. **Related Work.** The limitations of the traditional programming methods have encouraged the appearance of numerous researches on the development of HRI methods for robot programming. HRI can be upgraded and improved using Augmented Reality (AR) techniques. AR is a term used for overlaying computer-generated graphics, text and Three Dimensional (3D) models over images of the real scene. AR-based human-robot interfaces permit the operators to visualize both the virtual information and real-world environment simultaneously. The virtual elements represent visual cues and enhancements for a better understanding of the environment specially in applications such as maintenance [24], assembly [25], path planning [26], surgeries [27], and teleoperations [28].

Projector-based AR interface has been utilized in industrial robotics programming [10,29,30]. In this technique, the operator-defined key-points, paths, and different task parameters are projected on the workpiece's surfaces. The technique, although very intuitive, has two major limitations. First, the augmented graphics can only be projected on 2D surfaces. This hinders the use of the technique in planning 3D paths in space. Second, the visual information that can be provided this way is usually limited. For instance, robot end-effector 3D position and orientation cannot be provided. Consequently, an additional screen is needed to visualize the missed information. Getting data from different sources is not recommended in most cases, as the operator has to mentally fuse the information.

Several researches have been conducted on AR-assisted robot programming techniques in which operators can move freely in the work cell holding smartphones [31] and tablets [32]. These approaches give operators a good view of the AR domain and make it easier for them to correlate their motions. Other researches use handheld devices such as laser pointer [33] or a fiducial marker [22] to define robot end-effector path. In [22], the developed programming method merges all AR overlays in one screen. The system uses an overhead camera at the work cell in order to provide the operator with an overview of the robot scene. The camera is also used to track a handheld fiducial marker in the workspace. A questionnaire survey was done by the authors where candidate participants were given two programming tasks: pick-and-place and robot path-planning. The task completion time was measured for every participant and they were given questions to express intuitiveness and convenience of the whole system. The technique was effective and easy to use for operators with little programming background. However, researches such as [22,31-33] depended mainly on the user existence in robot environment. It is worth mentioning that, this is not allowed in many circumstances for safety reasons.

To avoid existence in the robot workspace for programming, Ni et al. [34] developed a double-module AR interface. The interface consists of a visual AR interface and a haptic interface. The visual AR interface is responsible for providing a display for a virtual robot motion augmented over a real-time video captured by a remote camera in the workspace. The operator can move the virtual robot using the haptic interface. Depending on the distance between the virtual robot end-effector and the workpiece, the operator gets a force feedback that helps him to maintain a constant distance to the workpiece surface. The system was successfully implemented in a welding process. The results showed that the system was intuitive and user-friendly. The technique proposed in [34], although being user-friendly for the specific task defined above, may not be the same for more complex tasks. For instance, the operator may be asked to remotely program a robot to pick an object in a certain pose, move it through a collision-free path in crowded environment, and place it correctly in a specific pose. In such cases, depending only on the operator's ability to remotely determine objects' position and orientation is not a good choice. It is better if the operator gets some assistance by automatically determining objects' poses and possibilities of collision.

To overcome drawbacks and limitations of the programming techniques presented above, this paper presents an intuitive user-friendly offline programming method to facilitate programming a remotely located robot in unstructured environment. The methodology exploits operator's intuitiveness in identifying objects and obstacles and provides him with an AR-assisted programming environment. Utilizing robot kinematics and transformation matrices, a volume of space around robot links is considered to detect expected collisions and to ensure safe task implementation. Vision algorithms and solid geometry formulas are fused to remotely estimate position and orientation of objects in workspace. Robot possible collisions, workspace boundaries, identified objects' poses, and smoothed target paths are all reflected to the AR environment. Experiments are conducted to validate the system where time of programming and success rate are recorded.

3. System Architecture.

3.1. **System hardware setup.** The system hardware illustrated in Figure 1 consists of a 6-DOF arm robot with 6-revolute joints and a gripper attached at the end of the robot arm to facilitate moving and manipulating small-size objects. Two fixed LifeCam HD Microsoft cameras (1080p full HD) are mounted at the remote working environment. A PC of (Corei7-4510U processor, 16GB RAM and NVIDIA GeForce GTX 1050 graphics card) is used to run vision algorithms. Finally, a microcontroller type Arduino board based on ATmega2560 microprocessor is used to generate the required motion control signals.

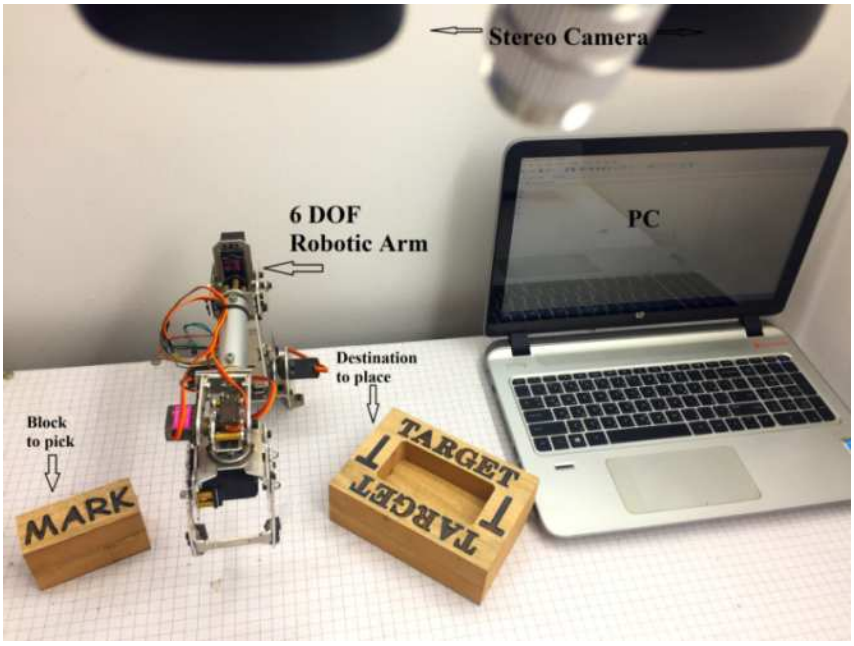


FIGURE 1. Overview of hardware components

3.2. **System software overview.** Figure 2 shows an overview of the system architecture where the operator interacts with the remotely installed 6-DOF robot. At the remote side, images are captured using stereo camera and sent to the PC in the operator side. Interaction starts between the operator and the real scene images displayed on the PC monitor. The operator selects objects, path points, and destination which are fed to the task planner. The latter uses vision algorithms to determine objects' position and orientation and the required motion through the required path. Collision detection algorithm

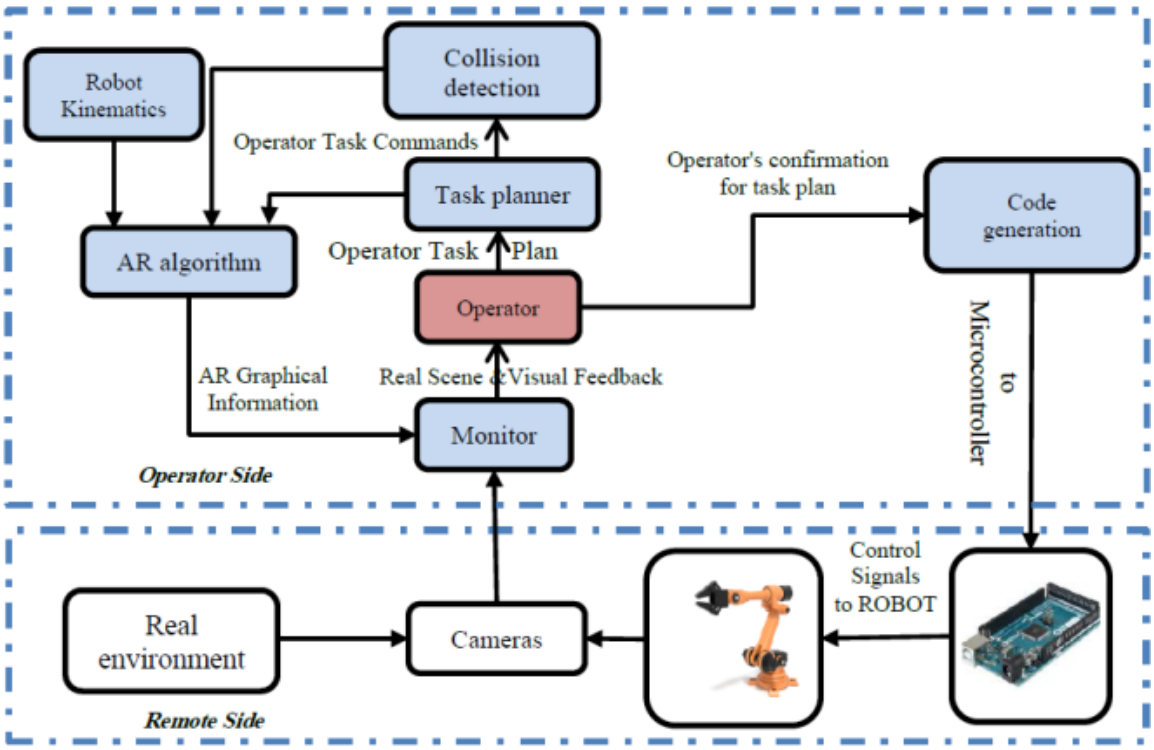


FIGURE 2. Overview of the proposed AR system architecture

detects whether the planned task will be implemented safely or collisions may happen. AR algorithm receives data from the task planner collision detection algorithm, and robot kinematics and converts it to virtual graphics superimposed on the image of the real environment. The operator is kept in a loop of task modification and AR-feedback. The task is only executed when there is no modification needed and the operator confirms execution. Finally, the operator's commands are converted into a control code which is sent to the robot's microcontroller. Unlike the similar researches presented in Section 2 the proposed AR system has the following distinctions.

- No marker cube or pointer is needed to teach the robot; therefore, the existence of the operator in robot environment is not required. This is in contrast to algorithms presented in [22,23], and [33].
- Operator gets assistance by automatically determining objects' 3D-pose which helps him accomplish complex tasks, as opposed to the algorithm presented in [34].
- No simulation of the whole robot is needed. Instead, a simple and perceptive simulation of the free-of-collision smoothed path is used.
- Finally, the algorithm takes account of the possible collision with the whole robot body, not only its EE. This is of potential importance, especially in unstructured environment.

4. **Methods and Materials.** Most of the tasks conducted by the industrial robots can be split into either one or an integration of pick-and-place and path-following tasks. This research focuses on the intuitive planning of robotic pick-and-place tasks, while considering the generation of a free-of-collision path in unstructured environments. The necessity of the user absence from the workspace during robot programming is emphasized. An AR-based programming interface is proposed for unskilled users to simplify offline robot programming. The method reduces mental fatigue and cognitive load to allow for intuitive representation and analysis of the data from the remote workspace. Stereo vision is

used to send information from the remote environment to the user for robot task planning and programming. Camera calibration allows for accurate overlaying of virtual graphics on the real scene image. A monitor enables the user to perceive the virtual cues for perceptive interaction and gives the user the feel of realism. Stereo matching algorithms modified by estimated homography for matching correspondences between the stereo images. Euclidean distance and cosines law equations are used to calculate object position and orientation. Path points are selected and interpolated for robot path following. Robot EE and links are taken into consideration while checking the selected path for possible collision with surrounding obstacles. The user is fed with visual feedback assisted by AR to evaluate the quality of the intended task. The proposed system provides flexibility of task modification in case the simulation is not satisfactory and visually warns the user for possible collision. Finally, the planned task is converted into controller code for task implementation. The detailed graphical cues are described in detail as follows.

i) 2D workspace boundary. The workspace of a robot is defined as a volume of space that the robot can reach in at least one orientation. As can be seen in Figure 3, the 2D working boundary (2D region of the robot workspace at the ground level) assists the user to insert the control points within the workspace. The boundary is measured relative to the robot base and the values are given to the AR algorithm as input.

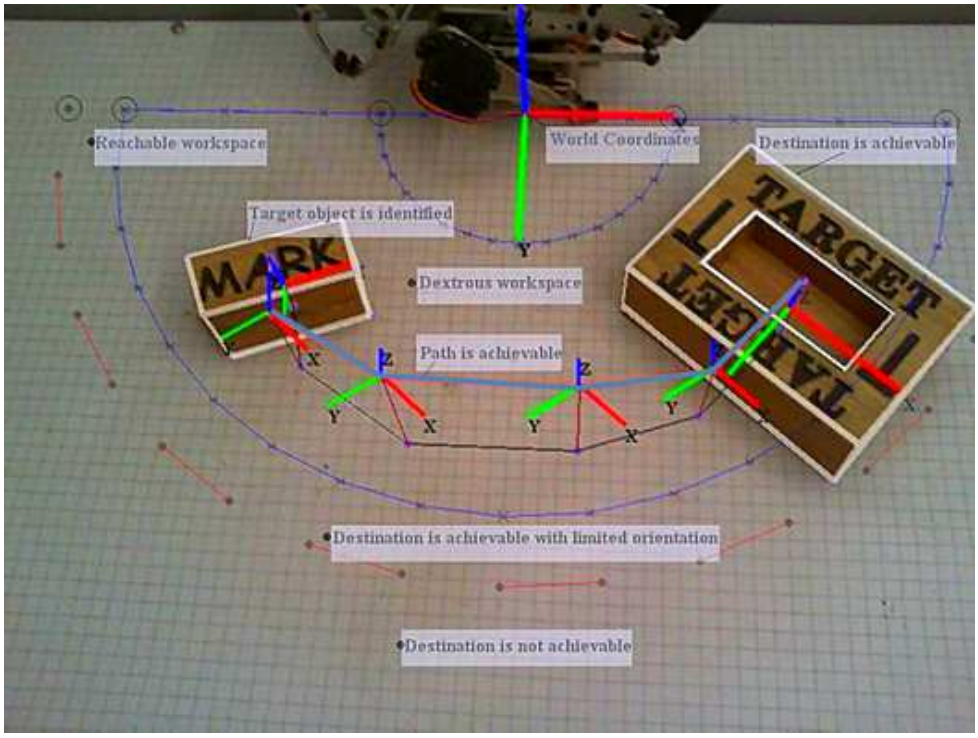


FIGURE 3. Operator programming environment

ii) Coordinate frame. As seen in Figure 3, some important frames are defined and displayed for easier perception of the spatial environment. This includes, the universal coordinate frame defined on the robot base, start and goal frames (for pick-and-place tasks), path points frames, (for path following tasks), and object frame defined at the target object's centroid. Euclidean distance is proposed to define the centroid of objects and provide the AR algorithms with important data for graphical representation.

iii) Object boundary. When an object is defined, a virtual cuboid (see Figure 3) is superimposed on the object to indicate that the object is correctly defined. The augmentation is based on the calculations of the object position and orientation.

iv) Rendering of paths. The path is formed by a series of spatial points that are selected on the ground of the workspace and given a specified height for each of them. The path is rendered using the equations of generated path.

v) Exception notification. During task planning, the path is augmented in red or blue color which means that an expected collision will or will not occur respectively. In case of expected collision, a message will appear referring to the robot link that is about to collide with a nearby obstacle. This guides the users to perform spatial point modification of this path segment to obtain a collision-free path, see Figure 3.

4.1. **Vision algorithms.** As has been mentioned before, it is difficult for the operator to accomplish complex tasks at distance in unstructured environment without getting a computer vision assistance. Vision algorithms assist operator to select objects and determine their 3D position and orientation. This is done through a sequence of three major processes, namely, the matching process, homographies estimation, and triangulation. Figure 4 depicts a block diagram of the developed algorithm steps for depth estimation. The following illustrates each process in detail.

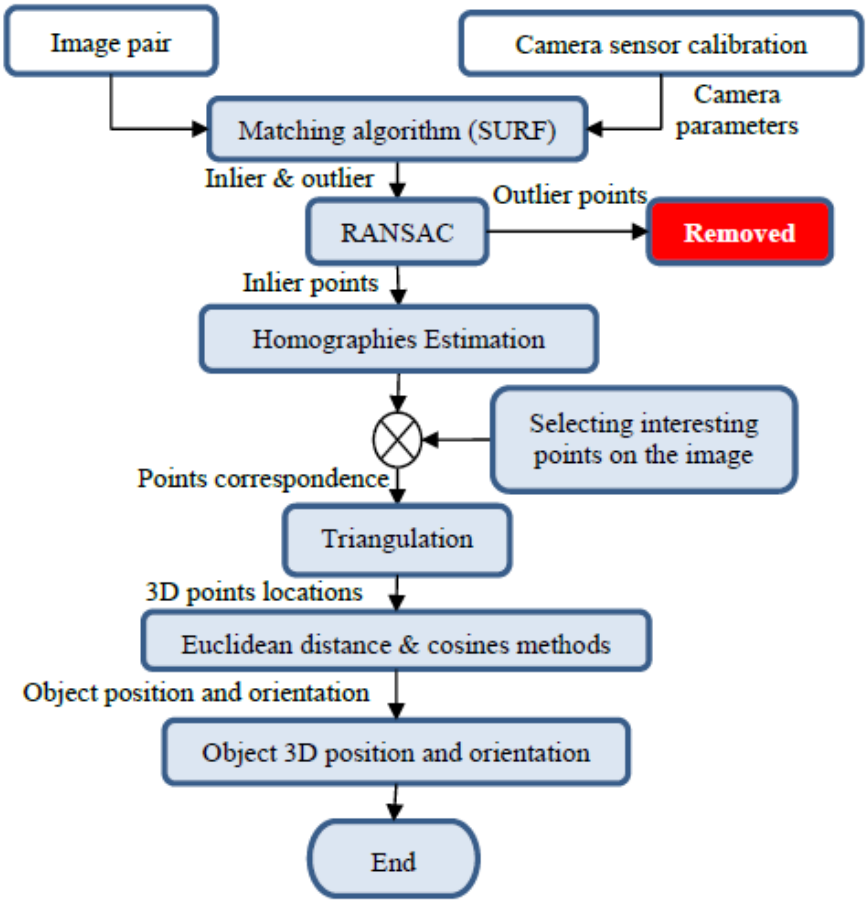


FIGURE 4. Block diagram of algorithm steps using for depth estimation

The matching process is a critical step in determining the accuracy of 3D pose estimation. SURF (Speeded Up Robust Features) is a well-known algorithm for feature extraction and matching [35]. RANSAC (Random Sample Consensus) is usually applied to excluding outliers (wrong matches) obtained by SURF and hence increases the accuracy of the obtained matched features [35]. Figure 5 shows the application of the SURF and RANSAC algorithms to the workspace images captured from the stereo camera. The

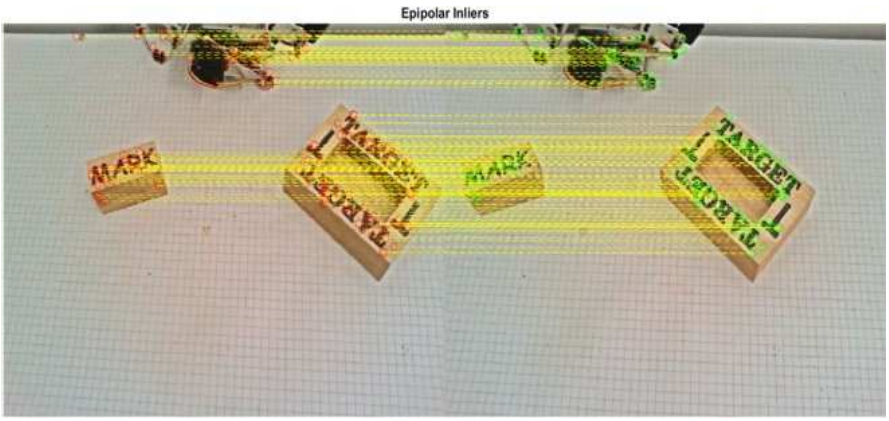


FIGURE 5. Matched points using SURF and RANSAC algorithms [36]

problem of using the SURF technique only is the clutter output of extracted features, which means that the user is dictated with the recognized matched features. The algorithm is modified to overcome this problem and facilitate finding correspondence of any point on the image that could not be described and matched using SURF. First, projective transformation matrices are estimated for any interesting plane of the target object. This can be represented as:

$$\boldsymbol{x_2} = \boldsymbol{H}\boldsymbol{x_1}$$

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{1}$$

where (x_1, y_1) and (x_2, y_2) are a pair of matched points on the left and right images respectively.

In order to estimate the matrix ($\boldsymbol{H}$), data of at least four pairs of points on the interested plane are required [36]. These are obtained from the matched pairs resulted from SURF and filtered by RANSAC. Figure 6 shows the identified top matched plane of the destination place. Now, any point in this plane can be matched to the corresponding point in the other images.

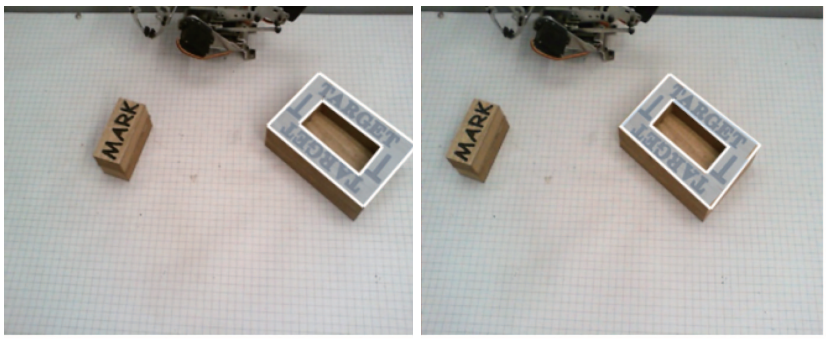


FIGURE 6. Identified top matched plane of the destination place

In order to determine the centroid position of an object in 3D space relative to the world coordinates (robot base), first, a cuboid is built around the object, as illustrated in Figure 7. Then, triangulation (back-projection) is applied to determining the 3D locations of the cuboid vertices according to Equation (2).

$$\underline{\boldsymbol{x}} = \boldsymbol{P}\underline{\boldsymbol{X}}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \times \begin{bmatrix} P_1^T \\ P_2^T \\ P_3^T \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = 0 \tag{2}$$

where ($\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{X}}$) are 3×1 vectors of point locations in the image and the world coordinates respectively, and P_i^T is the i-th row of the camera parameters matrix $\boldsymbol{P}$. Finally, Sampson correction, Equation (3), is applied to compensating the imperfection of matching correspondences, minimizing the geometric error and satisfying the epipolar constraint.

$$\begin{bmatrix} \hat{x}_1 \\ \hat{y}_1 \\ \hat{x}_2 \\ \hat{y}_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{bmatrix} - \frac{\boldsymbol{x'^T F x}}{(\boldsymbol{F}\underline{\boldsymbol{x}}_1)_1^2 + (\boldsymbol{F}\underline{\boldsymbol{x}}_1)_2^2 + (\boldsymbol{F}\underline{\boldsymbol{x}}_2)_1^2 + (\boldsymbol{F}\underline{\boldsymbol{x}}_2)_2^2} \begin{bmatrix} (\boldsymbol{F^T}\underline{\boldsymbol{x}}_2)_1 \\ (\boldsymbol{F^T}\underline{\boldsymbol{x}}_2)_2 \\ (\boldsymbol{F}\underline{\boldsymbol{x}}_1)_1 \\ (\boldsymbol{F}\underline{\boldsymbol{x}}_1)_2 \end{bmatrix} \tag{3}$$

where $(\hat{x}_1, \hat{y}_1)$, $(\hat{x}_2, \hat{y}_2)$ are point coordinates on the left and right images after correction and $\boldsymbol{F}$ is the fundamental matrix, $\underline{\boldsymbol{x}}_1$ and $\underline{\boldsymbol{x}}_2$ are 3×1 vectors of a point location on the left and right images respectively, and $(\boldsymbol{F}\underline{\boldsymbol{x}})_j$ is the j-th entry of vector $\boldsymbol{F}\underline{\boldsymbol{x}}$.

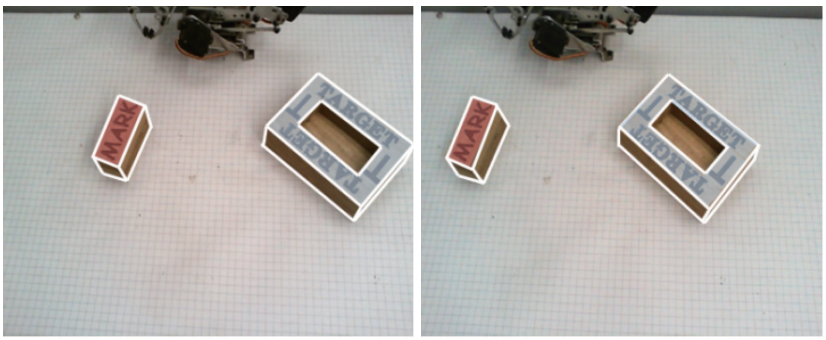


FIGURE 7. Determination of an object's position and orientation

Euclidean distance is then applied on the cuboid vertices to estimating its dimensions and centroid in 3D.

The object's orientation is estimated using the cosine law as:

$$\cos \alpha_{ij} = \frac{\boldsymbol{X}_{Ti} \cdot \boldsymbol{X}_{Wj}}{|\boldsymbol{X}_{Ti}| \cdot |\boldsymbol{X}_{Wj}|}, \tag{4}$$

where $\boldsymbol{X}_{Ti}$ represents the target coordinates ($i = 1, 2, 3$) and $\boldsymbol{X}_{Wj}$ represents the world coordinates ($j = 1, 2, 3$).

For AR study, all estimated points are projected on the image using camera forward projection, Equation (5).

$$\begin{gathered} \lambda \boldsymbol{p} = \boldsymbol{K}\boldsymbol{R} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} - \boldsymbol{K}\boldsymbol{R}\boldsymbol{C} \\ \lambda = \frac{Z - C_z}{Z_3}, \text{ where } (Z_1, Z_2, Z_3)^T = \boldsymbol{R^{-1}}\boldsymbol{K^{-1}}\boldsymbol{p} \\ \boldsymbol{C} = (C_x, C_y, C_z)^T \end{gathered} \tag{5}$$

where $\boldsymbol{C}$ and $\boldsymbol{R}$ are the translation and rotation matrices between camera and world coordinates respectively, $\boldsymbol{K}$ is the camera intrinsic parameters, $\boldsymbol{p}$ is the pixel point, $\boldsymbol{p} = (x, y, 1)^T$, and $(X, Y, Z)^T$ is a 3D world homogeneous point.

4.2. **Collision-free path.** The obstacle-avoidance problem usually defines how to control the manipulator in order to track the desired end-effector trajectory while simultaneously ensuring that no part of the manipulator collides with any obstacle. In order to avoid any possible collision, the manipulator has to move away from obstacles in a configuration where the distance between them is large enough for avoidance. Unstructured environments pose an additional challenge in automatic robot programming methods. The methodology proposed in this research exploits operator's intuitiveness in identifying obstacles and provides him with the assisting AR tools to accomplish a free collision task planning. Figure 8 depicts the methodology of collision-free approach used in the current research. First, all obstacles are identified by the operator the same way as discussed before for the target object and destination. After that, the operator's initial task planning is used through the robot inverse kinematics to solve for joints parameters converting the Cartesian space into joint space. Interpolation is used to get values of intermediate points. Robot forward kinematics is then used to calculate for the joints 3D position into Cartesian space. These positions are used to create virtual cuboid around each robot link, see Figure 9. The dimensions of each cuboid depend on the robot physical dimensions and allow for a suitable safety margin. A checking algorithm is then applied to checking for possible collision between robot physical body and obstacles. This is done by checking 3D occupied volume overlap for both robot physical body and identified obstacles. Finally, if a potential collision is detected, the operator is notified and asked for a path modification.

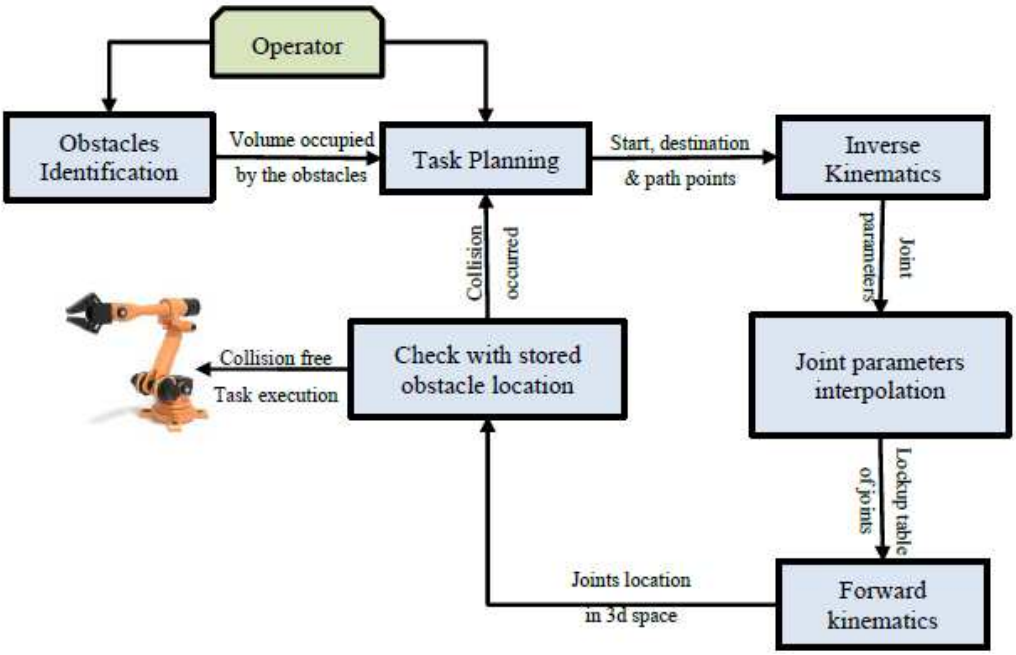


FIGURE 8. Methodology of collision-free approach

4.3. **Interactive path planning.** In order to increase the robot life time and prevent jerking at the selected control points, a cubic-spline interpolation is used to generate a second order polynomial path that is continuous at these control points. According to [37], the cubic-spline interpolation is described as

$$C_i(x) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{6}$$

where (a_0, a_1, a_2, a_3) are four parameters that can satisfy the required spline. The method is adopted here in order to smooth the path line ordered by the operator. Figure 10

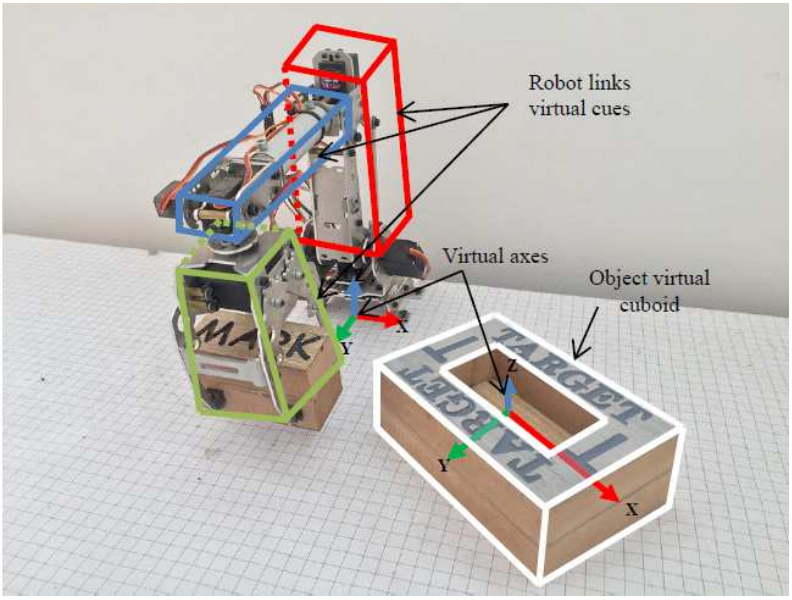


FIGURE 9. Overlaying virtual cues on real world

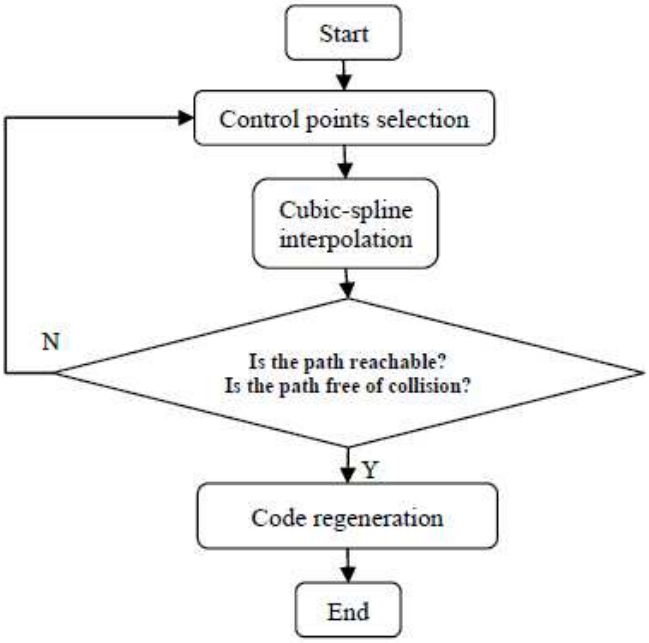


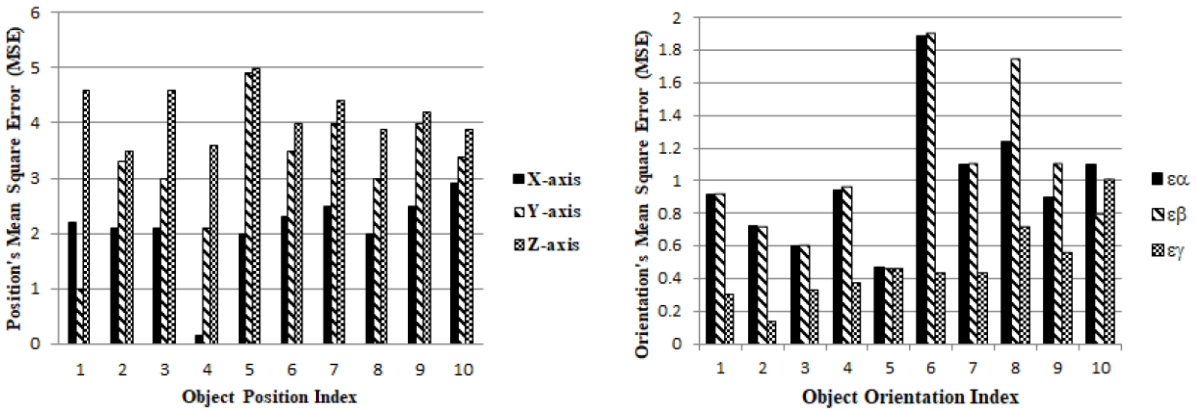
FIGURE 10. Path generation procedure

describes the path generation procedure. After determining the spline parameters, robot kinematics is used to check the generated path for reachability and collision possibility. In case the generated path is not satisfactory, or a collision is detected, the operator is asked for a modification.

5. **Experimental Work and Results.** Experiments are conducted using the system described in Section 3.1 to evaluate the proposed system performance. First, the accuracy of the vision algorithms in estimating object's position and orientation is evaluated. Then, a pick-and-place task is designed where the operator is asked to pick an object, choose a free-of-collision path, and place it in a certain pose. Since some of the system performance parameters depend on a qualitative human assessment, it is better to evaluate it based on

different users' opinions. This includes degree of intuitiveness, ease of use, convenience, and the quality of the virtual feedback. For this purpose, a questionnaire is prepared on the system behavior and a group of candidate participants is invited to conduct the experiments and then answer the questionnaire.

5.1. **Vision algorithm accuracy.** The experiment aims at evaluating the accuracy of the proposed vision algorithm. The target object has been randomly set in predefined positions and orientations relative to the world coordinates (reference). The algorithm was run, and the estimated position and orientation were compared to the actual measurements. The position error was defined as the difference between the estimated and actual positions of the object centroid, while the orientation error was defined in terms of error angles of rotation (ε_α, ε_β, ε_γ) between the estimated and actual axes (X, Y, Z). Figure 11 shows the estimation errors recorded for ten randomly selected poses. The maximum recorded absolute errors for position were (2.9 mm, 4.9 mm, 5 mm) and the RMSEs (Root Mean Square Errors) for all measurement points were (2.18 mm, 3.37 mm, 4.19 mm). In the same manner, the maximum recorded absolute error angles were (1.89°, 1.9°, 1.01°), while the RMSEs were (1.05°, 1.12°, 0.53°). It is worth mentioning that, the level of accuracy of the used vision algorithm is found to be acceptable in a manner that it will not hinder the investigation of the proposed HRI. Recorded errors in similar researches, where different tools were used to define robot path, were 11 mm (marker-based interaction) [23], and 15 mm (haptic-based interaction) [34].



(a) (b)

FIGURE 11. Error estimation for ten randomly selected object's poses: (a) position (mm); (b) orientation (degree)

5.2. **Pick-and-place task.** A case study is performed to evaluate the performance of the proposed system. First, the operator uses the provided image and a PC input device (mouse) to select target object, obstacles and target destination the same way as discussed in Section 4. A cuboid is drawn around the selected object to confirm the operator choice. Then, the required path is selected between the object and the destination by picking some waypoints inside the robot workspace. The operator is alarmed if any of the path points is outside the robot reachable workspace. For the sake of clarification, the task here is implemented in two scenarios. The first scenario is when a potential collision is expected, see Figure 12. In this case, the path is colored in red; the operator is informed of the robot link to be collided and asked to enter another path. The second scenario is when there is no potential collision with any of the existing obstacles, see Figure 13. In this case, the

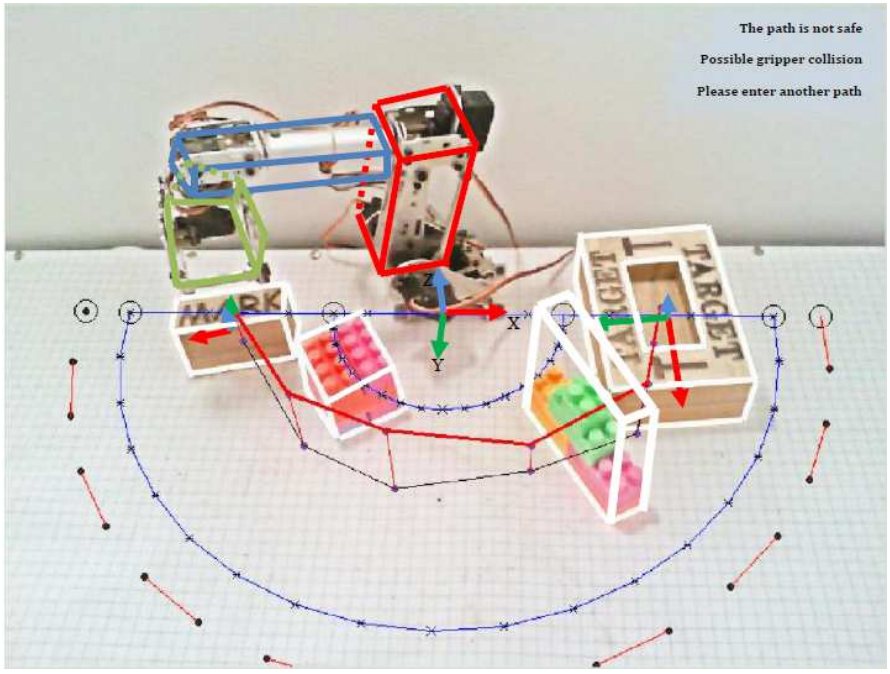


FIGURE 12. (color online) Expected collision scenario

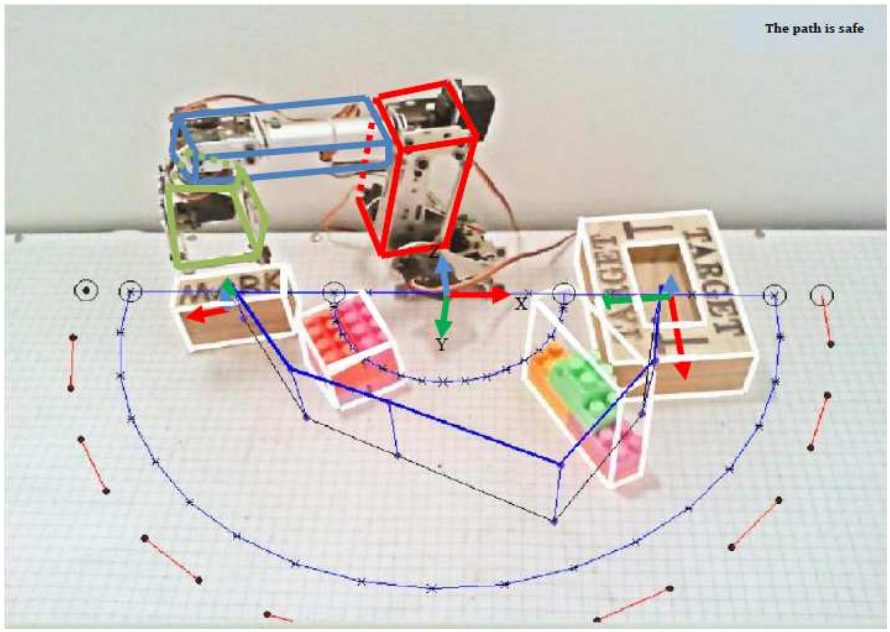


FIGURE 13. (color online) Collision-free scenario

required path is converted into executable code using robot IK as discussed before. Figure 14 shows the planned path using 6 control points where the red and blue lines represent the path before and after modification respectively. In the latter case the operator has changed the location of point-3 only to avoid collision.

Figures 15(a)-15(h) show the robot motion sequence for the designed task. As can be seen from Figure 15(a) and Figure 15(b) the robot arm moves and adjusts the end-effector orientation to the object orientation. In Figures 15(c)-15(g) the arm picks the object, elevates it, adjusts orientation and moves to the destination following the designed path. Finally, in Figure 15(h) the arm returns back to the home position.

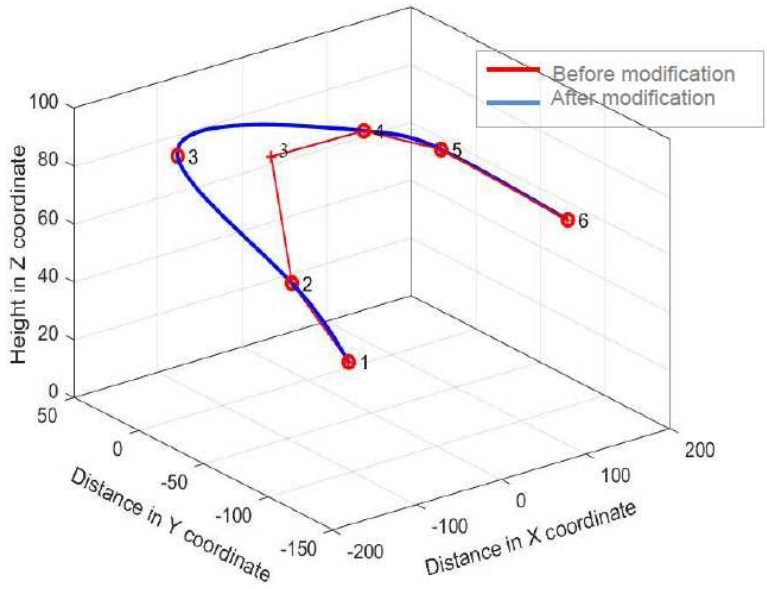


FIGURE 14. (color online) Robot EE path before and after modification

The same task has been given to 5 candidate participants (with mechanical engineering background). A questionnaire is prepared on the participants' opinion and suggestions on the proposed system. The users' trials are evaluated using a 5-point Likert scale (5: strongly agree; 4: agree; 3: neutral; 2: disagree; 1: strongly disagree). The questionnaire contains the following items.

a) Robot is easy to program.
b) Pick-and-place tasks are easy to program using the system.
c) The proposed system is intuitive to use.
d) The proposed system is intuitive to learn.
e) I feel confident that the intended task will be implemented correctly.
f) I feel confident that the intended task will be safely implemented.

Figure 16 shows the time required by 5 participants to complete the pick-and-place task using the proposed system. The questionnaire results' analysis showed that, all the participants were able to remotely interact with the working environment using the proposed system. They felt intuitive and convenient to carry out the operations of defining objects, destination, and the EE path. With regards to the visual cues and feedback presented on the monitor screen, the participants rated that it helped them understand and obtain the feeling of the intended path. The location of the object and destination were estimated with accepted accuracy and emphasized by the virtual feedback. However, it was reported that, the virtual feedback of the orientation was not expressive enough. Moreover, operators found difficulty in defining obstacles of irregular shapes compared with regular ones.

6. **Conclusions and Future Work.** In this study, an AR-based interface for intuitive HRI is proposed to assist inexperienced users in offline programming of remotely-located robots in unstructured environments. This has special importance in uncomfortable and hazardous environments where the operator does not exist in the working environment. Safety of task implementation is taken into consideration where the user is warned for potential arm collision before execution. A modified matching method is developed to solve the drawback of limited matched features in the conventional methods SURF and

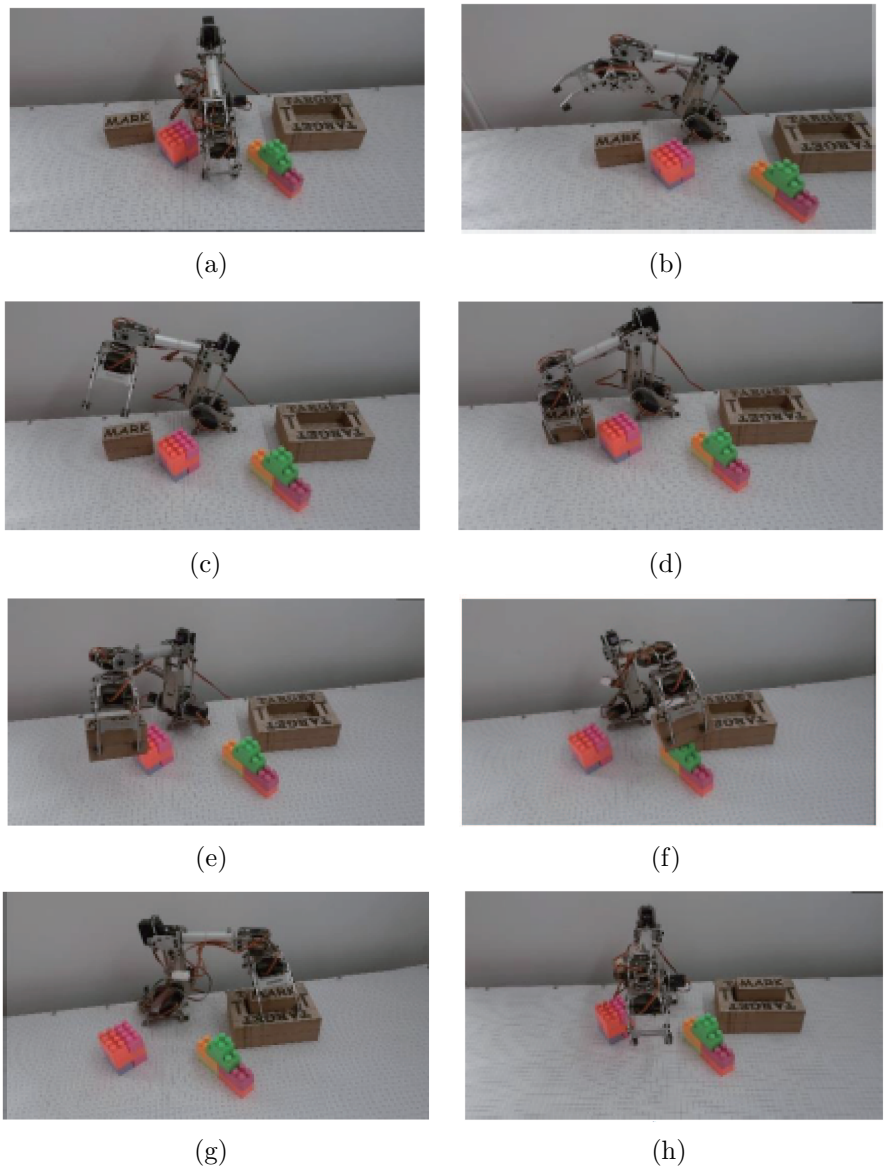


FIGURE 15. Robot motion sequence while accomplishing the pick-and-place task

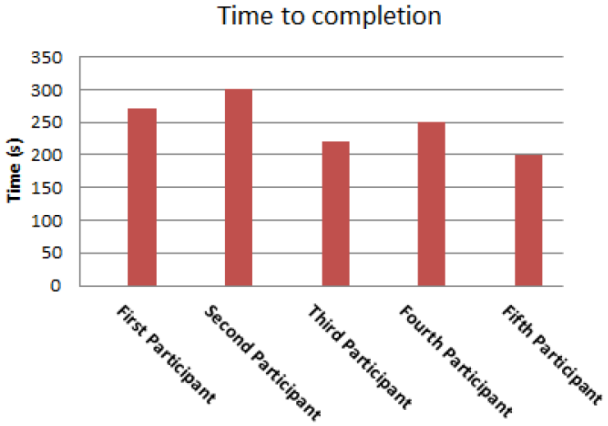


FIGURE 16. Time for the participants to complete the pick-and-place task

RANSAC. A test-rig was set up to validate the proposed system. According to the questionnaire survey, the candidate operators reported that the AR interface was user-friendly. It was easy and intuitive to select the control points and define the position and orientation of target objects, destination and obstacles. Operators felt that the simulation and visualization through the AR interface of the intended task increased their confidence that the planned task would be correctly and safely performed. However, they have reported that the virtual feedback of the orientation was not expressive enough. They found difficulty in defining obstacles of irregular shapes compared with regular ones. The overall results were very encouraging which indicates that the proposed approach is effective and intuitive in remotely planning robot tasks in unstructured environments. In future consideration of this work, a number of areas can be further developed to improve the overall performance of the system. This includes improving the virtual feedback of the EE orientation, methods of defining irregular objects, and developing algorithms in which the system suggests alternative collision-free paths to the users.

REFERENCES

[1] S. Ong, J. Chong and A. Nee, A novel AR-based robot programming and path planning methodology, *Robotics and Computer-Integrated Manufacturing*, vol.26, no.3, pp.240-249, 2010.

[2] J. O. Oyekan et al., The effectiveness of virtual environments in developing collaborative strategies between industrial robots and humans, *Robotics and Computer-Integrated Manufacturing*, vol.55, pp.41-54, 2019.

[3] W. Budiharto, E. Irwansyah, J. S. Suroso and A. A. S. Gunawan, Design of object tracking for military robot using PID controller and computer vision, *ICIC Express Letters*, vol.14, no.3, pp.289-294, 2020.

[4] A. Solyman, M. Roman, A. Keshk and K. Sharshar, Vision control of 5-DOF manipulator for industrial application, *International Journal of Innovative Computing, Information and Control*, vol.12, no.3, pp.735-746, 2016.

[5] B. Akan, A. Ameri, B. Cürüklü and L. Asplund, Intuitive industrial robot programming through incremental multimodal language and augmented reality, *IEEE International Conference on Robotics and Automation*, Shanghai, China, pp.3934-3939, 2011.

[6] J. Wassermann, A. Vick and J. Krüger, Intuitive robot programming through environment perception, augmented reality simulation and automated program verification, *Procedia CIRP*, vol.76, pp.161-166, 2018.

[7] Z. Pan, J. Polden, N. Larkin, S. Van Duin and J. Norrish, Recent progress on programming methods for industrial robots, *ISR 2010 (The 41st International Symposium on Robotics) and ROBOTIK 2010 (The 6th German Conference on Robotics)*, VDE, pp.1-8, 2010.

[8] S. Deng, Z. Cai, D. Fang, H. Liao and G. Montavon, Application of robot offline programming in thermal spraying, *Surface and Coatings Technology*, vol.206, nos.19-20, pp.3875-3882, 2012.

[9] L. Qi, X. Yin, H. Wang and L. Tao, Virtual engineering: Challenges and solutions for intuitive offline programming for industrial robot, *IEEE Conference on Robotics, Automation and Mechatronics*, Chengdu, China, pp.12-17, 2008.

[10] G. Reinhart, U. Munzert and W. Vogl, A programming system for robot-based remote-laser-welding with conventional optics, *CIRP Annals – Manufacturing Technology*, vol.57, no.1, pp.37-40, 2008.

[11] G. Erdős, C. Kardos, Z. Kemény, A. Kovács and J. Váncza, Process planning and offline programming for robotic remote laser welding systems, *International Journal of Computer Integrated Manufacturing*, vol.29, no.12, pp.1287-1306, 2016.

[12] H. Fang, S. Ong and A. Nee, Interactive robot trajectory planning and simulation using augmented reality, *Robotics and Computer-Integrated Manufacturing*, vol.28, no.2, pp.227-237, 2012.

[13] A. Brunete et al., User-friendly task level programming based on an online walk-through teaching approach, *Industrial Robot: An International Journal*, vol.43, no.2, pp.153-163, 2016.

[14] F. Ferraguti, C. T. Landi, C. Secchi, C. Fantuzzi, M. Nolli and M. Pesamosca, Walk-through programming for industrial applications, *Procedia Manufacturing*, vol.11, pp.31-38, 2017.

[15] C. T. Landi, F. Ferraguti, C. Secchi and C. Fantuzzi, Tool compensation in walk-through programming for admittance-controlled robots, *The 42nd Annual Conference of the IEEE Industrial Electronics Society (IECON 2016)*, pp.5335-5340, 2016.

[16] L. Bascetta, G. Ferretti, G. Magnani and P. Rocco, Walk-through programming for robotic manipulators based on admittance control, *Robotica*, vol.31, no.7, pp.1143-1153, 2013.
[17] S. Choi, W. Eakins, G. Rossano and T. Fuhlbrigge, Lead-through robot teaching, *IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pp.1-4, 2013.
[18] V. Villani, F. Pini, F. Leali, C. Secchi and C. Fantuzzi, Survey on human-robot interaction for robot programming in industrial applications, *IFAC-PapersOnLine*, vol.51, no.11, pp.66-71, 2018.
[19] M. Ragaglia, A. M. Zanchettin, L. Bascetta and P. Rocco, Accurate sensorless lead-through programming for lightweight robots in structured environments, *Robotics and Computer-Integrated Manufacturing*, vol.39, pp.9-21, 2016.
[20] H.-C. Lin, *Embedding Intelligence into Robotic Systems – Programming, Learning, and Planning*, Ph.D. Thesis, UC Berkeley, 2018.
[21] G. Biggs and B. MacDonald, A survey of robot programming systems, *Proc. of the Australasian Conference on Robotics and Automation*, pp.1-3, 2003.
[22] H. C. Fang, S. K. Ong and A. Y. C. Nee, A novel augmented reality-based interface for robot path planning, *International Journal on Interactive Design and Manufacturing*, vol.8, no.1, pp.33-42, 2014.
[23] H. C. Fang, S. K. Ong and A. Y. C. Nee, Novel AR-based interface for human-robot interaction and visualization, *Advances in Manufacturing*, vol.2, no.4, pp.275-288, 2014.
[24] D. Aransyah, F. Rosa and G. Colombo, Smart maintenance: A wearable augmented reality application integrated with CMMS to minimize unscheduled downtime, *Computer-Aided Design and Applications*, vol.17, no.4, pp.740-751, 2020.
[25] Z. Wang et al., Information-level AR instruction: A novel assembly guidance information representation assisting user cognition, *The International Journal of Advanced Manufacturing Technology*, vol.106, nos.1-2, pp.603-626, 2020.
[26] C. Chen, Y. Pan, D. Li, S. Zhang, Z. Zhao and J. Hong, A virtual-physical collision detection interface for AR-based interactive teaching of robot, *Robotics and Computer-Integrated Manufacturing*, vol.64, 2020.
[27] N. Golse, A. Petit, M. Lewin, E. Vibert and S. Cotin, Augmented reality during open liver surgery using a markerless non-rigid registration system, *Journal of Gastrointestinal Surgery*, pp.1-10, 2020.
[28] J. D. Hernández, S. Sobti, A. Sciola, M. Moll and L. E. Kavraki, Increasing robot autonomy via motion planning and an augmented reality interface, *IEEE Robotics and Automation Letters*, vol.5, no.2, pp.1017-1023, 2020.
[29] D. Antonelli and S. Astanin, Qualification of a collaborative human-robot welding cell, *Procedia CIRP*, vol.41, pp.352-357, 2016.
[30] A. Gaschler, M. Springer, M. Rickert and A. Knoll, Intuitive robot tasks with augmented reality and virtual obstacles, *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, pp.6026-6031, 2014.
[31] J. Lambrecht and J. Krüger, Spatial programming for industrial robots based on gestures and augmented reality, *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.466-472, 2012.
[32] J. A. Frank, M. Moorhead and V. Kapila, Realizing mixed-reality environments with tablets for intuitive human-robot collaboration for object manipulation tasks, *Proc. of the 25th IEEE International Symposium on Robot and Human Interactive Communication*, NY, USA, pp.302-307, 2016.
[33] S. Ong, A. Yew, N. Thanigaivel and A. Nee, Augmented reality-assisted robot programming system for industrial applications, *Robotics and Computer-Integrated Manufacturing*, vol.61, 2020.
[34] D. Ni, A. Yew, S. Ong and A. Nee, Haptic and visual augmented reality interface for programming welding robots, *Adv. Manuf.*, no.5, pp.191-198, 2017.
[35] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, Speeded-up robust features (SURF), *Computer Vision and Image Understanding*, vol.110, no.3, pp.346-359, 2008.
[36] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2003.
[37] J. J. Craig, *Introduction to Robotics, Mechanics and Control*, 3rd Edition, Pearson Education International, 2005.