Assembly Assisted by Augmented Reality (A^3R)

Jun Okamoto Jr. and Anderson Nishihara

Abstract Traditionally, assembly instructions are written in the form of paper or digital manuals. These manuals contain descriptive text, photos or diagrams to guide the user through the assembly sequence from the beginning to the final state. To change this paradigm, an augmented reality system is proposed to guide users in assembly tasks. The system recognizes each part to be assembled through image processing techniques and guides the user through the assembly process with virtual graphic signs. The system checks whether the parts are properly assembled and alerts the user when the assembly has finished. Some assembly assisted by augmented reality systems use some kind of customized device, such as head mounted displays or markers to track camera position and to identify assembly parts. These two features restrict the spread of the technology whence, in this work, customized devices and markers to track and identify parts are not used and all the processing is executed on an embedded software in an off-the-shelf device without the need of communication with other computers to any kind of processing.

1 Introduction

In assembly tasks several elements need to be addressed such as the part's orientation and the assembly sequence. As a way to assist assembly tasks, several authors have proposed augmented reality systems [1–3]. They proposed that images augmented with instructions may be an alternative to the traditional paper manual and it would improve the assembly time. Implementations of augmented reality systems were compared to computer assisted instructions, paper manuals and tutorial by an expert. Although augmented reality (AR) systems did not improve assembly time compared to a tutorial by an expert [2], it was evidenced that assembly time can be improved

J. Okamoto Jr. (✉) · A. Nishihara
Department of Mechatronics and Mechanical Systems Engineering,
Escola Politécnica of the University of São Paulo, São Paulo, Brazil
e-mail: jokamoto@usp.br

A. Nishihara
e-mail: anderson.nishihara@usp.br

Y. Bi et al. (eds.), *Intelligent Systems and Applications*,
Studies in Computational Intelligence 650, DOI 10.1007/978-3-319-33386-1_14

in comparison to computer assisted instructions [1, 3] and paper manuals [1–3]. Yet, when an augmented reality system was used the number of errors in the assembly process was reduced [1, 3].

This research consists in development of an augmented reality system to assist the assembly process in real time as a replacement to conventional assembly manuals. The proposed system behaves as an interactive instruction manual where virtual models and graphic signs are overlaid on a video of the real scene so that the user is guided until the final assembled state is reached. To prove the concepts described here, a tablet is used between the user and the workspace in a way that a set of randomly placed assembly parts can be viewed on the display. From the video feed, the parts to be assembled are identified through image processing techniques and graphic signs show to the user where the piece should be assembled. Lastly, the system checks the assembly state and informs the user when the final state is reached.

The developed system has the following characteristics:

- Fiducial markers are not used: objects are detected by image processing and named from a database;
- The solution is known beforehand: the system is not for finding a solution to the assembly problem;
- The system does not teach the assembly process to the user, its purpose is to be a guide through the assembly process;
- The display is placed between the user and the assembly workspace and it is fixed during all the process;
- A general assembly guidance process is proposed and validated;
- The tablet implementation runs an embedded software, only resources from the tablet are used and no network connection is required.

This paper presents the results obtained by the object recognition subsystem and the assembly guidance process of the developed system that is being called Assembly Assisted by Augmented Reality (A^3R).

2 Augmented Reality in Assembly Tasks

One of the first efforts to build an AR tool was shown in [4] to guide the assembly of cable harnesses. Information derived from engineering design of parts and processes come to the factory floor in the form of templates, assembly guides, cable lists and location markings. Expenses and delays on manufacturing can be minimized if changes on engineering design of parts and processes can be quickly mirrored to templates, assembly guides, cable lists and location markings. To address this issue, an AR system composed of a wearable computer and a head mounted display (HMD) is demonstrated in [4]. Later, researchers demonstrated applications to guide printer maintenance [5] and car door-lock assembly [6].

Specifically on AR applications in assembly processes, we can analyze the research in this field under 4 main aspects which enables the AR system implementation: display, type of tracking system, user interaction and platform. Additionally, an assembly sequence representation that is needed to guide the user through the assembly process is also analyzed.

2.1 Display

Basically, there are two different approaches to show information to the user in AR systems: optical see-through and video see-through. In the optical see-through approach a half-transparent mirror is placed in front of the user's eyes. In this way the world can be seen though the half-transparent mirror and information is reflected on these half-transparent mirrors, thereby combining the real world and virtual information. Optical see-through displays do not present parallax effect, however, the combination of virtual elements by half-transparent mirror can reduce the brightness and contrast of virtual images. In the video see-through approach the user sees the world captured by cameras and information is superimposed on the digitized video. Video see-through approaches are cheaper and easier to be implemented. Besides, it is easier to add or remove elements from the display, as the world is already digitized. Still, as the camera and user's eyes have different field of views, video see-through approaches suffer from parallax effect. As for the placement of the display, some devices are placed on the user's head and the display is positioned in front of the eyes (HMD—head mounted displays), projectors can be used to display information on real objects (spatial displays) and some displays are placed on a hand's reach (hand-held displays).

On the literature, optical see-through [3, 7] and video see-through HMD [1, 7–13] are commonly used in AR systems as they enable the use of both user's hands on the assembly process. However, HMDs available on the market are bulky, expensive or requires connection to a computer.

Another approach is the use of spatial displays that use projectors to directly display information on real objects. This is the most integrated technology to the environment and several users can interact simultaneously. Projector-based AR systems implementations are shown in [14–16]. However, the only use of a projector as a spatial display to guide the user to a solution of a 3D puzzle is presented in [15]. Although spatial systems enable more immersion on the task than other alternatives, it requires a controlled environment illumination and preparation of workspace parameters to correctly superimpose information on assembly parts.

Smartphones, tablets and PDAs represent hand-held displays. A system composed of a smartphone and PC to guide an assembly process of a 3D puzzle is shown in [17]. While the PC made all the image processing, the smartphone was used to display images augmented with assembly information.

2.2 Tracking

When augmented reality is used only to show information to the user and it is not required to superimpose information aligned to real world, this is called augmented reality without context as demonstrated in [3, 8, 15, 18]. In this kind of AR, images, animations and texts are shown to the user on a display similarly to an electronic manual. The main difference between an electronic manual and AR without context is the faster access to assembly information without change of attention, as the information is in the same display as the real world seen by the user.

Augmented reality with context requires the knowledge of camera and object position. In some cases, camera and object position are not known beforehand, therefore, methods to extract this information are needed. These methods are often called tracking. The main challenge to tracking methods is the depth information which is lost when a camera captures the world. Some AR systems use a combination of camera and another sensor to acquire the depth information. Magnetic sensors [1], infrared sensor [19] and multiple cameras [20] have been used to extract the depth data.

Specifically on AR uses on assembly processes, the main approach to tracking is the application of computer vision techniques on images taken by a singular camera. Two different approaches using computer vision techniques are found on the literature: tracking with fiducial markers and marker-less tracking. Fiducial markers make the tracking and recognition more robust, however, it requires the preparation of the workspace. AR systems which use fiducial markers tracking approach are presented in [9–12, 16, 17, 21–23].

There are mainly two kinds of marker-less tracking: frame-to-frame and tracking by detection. Frame-to-frame methods estimate the current position by using information from previous frames. Tracking by detection estimates position by matching extracted features from images to features extracted from a known object model. An AR system that uses tracking by combining frame-to-frame and tracking by detection methods was shown in [24]. While frame-to-frame tracking is achieved by an implementation of a particle filter algorithm, tracking by detection is made by matching features from several 2D synthetic views of the 3D geometric features generated by moving a virtual camera along a sphere.

2.3 User Interaction

Some researchers used mouse and PC keyboard [9, 10, 16] or smartphone keyboard [17] to enable the AR system interaction, while others proposed interfaces with selectable panels by using a 3D stick tool [21], hand and gesture recognition methods [12] and haptic interfaces [20].

Two kinds of tools are used to interact in AR environments: computer vision tools and haptic tools. A system controlled by hand gestures is demonstrated in [10]. The detection of hands is made by color segmentation and virtual menus are opened by

holding the hands on an activation area. A stick tool detected by image segmentation to interact with virtual panels is presented in [21]. Following the work in [21], the segmentation technique to recognize hands and gestures, respectively was improved in [25, 26].

An AR system with voice control was presented in [10, 27]. The voice control is made by a connection from the AR system with a stand-alone voice command system which makes the voice recognition of simple English commands like "next", "previous", "next phase" and "previous phase".

Yet, an AR system with a haptic device where users wear a device to manipulate virtual objects as they were real objects was demonstrated in [20], which enables a more integrated and natural way to interact with the virtual elements.

2.4 Platform

Several AR systems were implemented by a combination of PC and HMD/monitor [1, 3, 8, 10–16, 24, 27, 28]. The PC versatility, connection diversity which enables several devices connected at the same time, wide availability on market and processing power are the main reasons for choosing PCs as platforms for AR systems. Still, there are restrictions on connections to HMD, as a connection to a PC is required.

One main requirement on AR systems is near real time response. Manipulation of 3D virtual objects, tracking and object recognition require suitable processing power and memory. Due to this requirement, few approaches have been developed on mobile devices and markers for object recognition or tracking have been used to reduce the need of complex image processing algorithms. An AR system using a mobile phone as a client and a PC as a server on a client/server architecture is shown in [17]. A mobile phone is used to take photos of the workspace with markers and the images are sent to a server. The images are processed on the server and then the assembly information is sent back to the mobile phone. On the research presented in [24] a marker-less tracking subsystem for AR systems was described. Images at 640×480 resolution were used to estimate camera pose in hundreds of milliseconds on a PC equipped by Intel Core i7-860 CPU and 3GB of RAM. The development of an AR system with the same architecture as proposed in [17] was proposed in [24] to eliminate the need to move bulky hardware components so that the mobile device's computational requirements could be lowered. Up until this work, only one project implemented on a tablet [23] was found on the literature, mainly due to limitations on processing power and memory. Nowadays, mobile devices are becoming increasingly more powerful platforms. The iPad is a mobile device that stands out for its computing power, which can allow execution of image processing algorithms in real time and may be used for AR systems [29].

2.5 Assembly Sequence Representation

Some assembly sequence representations have been proposed in [30, 31]. Whenever an intelligent control system for assembly processes is being designed, a way to describe the assembly sequence representation is required. The choice of the representation impacts in the required memory to store the information and the difficult to derive the representation from an assembly process.

The following definitions are used for the assembly sequence representation:

- *Subassembly*—represents a set of joined parts, it can contain one or more parts
- *Assembly task*—is a connection between subassemblies
- *Assembly sequence*—ordered list of assembly tasks
- *Assembly state*—list of connected subassemblies
- *Assembly layout*—represents the part disposition in the space

The connection between solid parts forming an stable unit is a mechanical assembly. The assembly parts are connected by surface contact reducing the degrees of freedom. Subassemblies are sets of parts containing one or more joined parts. Whenever subassemblies are connected an assembly task is performed. A succession of an ordered list of tasks or assembly states is an assembly sequence. An assembly process starts with all parts separated and ends with all parts connected to form a whole object.

An assembly sequence representation named as Assembly Sequence Table (AST) was introduced in [31]. This representation is used to describe all possible solutions to an assembly problem for evaluation and selection of an assembly sequence. Table 1 shows an example of AST. The first column is the level number, the second column contains the assembly states representing all feasible subassemblies formed after the assembly task is performed. Assembly tasks represent the feasible subassemblies from previous level to be joined. At any level L the assembly states column have all the subassemblies with L number of parts. Hence at level 1 only subassemblies with 1 part on assembly states are listed, on level 2 there are 3 assembly states with sets containing 2 parts each and so on.

Table 1 AST example of parts identified by a, b, c and d

Level	Assembly states	Assembly tasks
1	{a}; {b}; {c}; {d}	–
2	{a, b}	{(a), (b)}
	{a, c}	{(a), (c)}
	{c, d}	{(c), (d)}
3	{a, b, c}	{(a, b), (c)};{(a, c), (b)}
4	{a, b, c, d}	{(a, b, c), (d)};{(a, b), (c, d)}

3 The Proposed System

The Assembly Assisted by Augmented Reality (A^3R) system was developed to run an embedded software on an off-the-shelf tablet with a camera, display, gyroscope and accelerometer sensors. It uses a video see-through approach to show the assembly space to the user and graphical signs to guide the user through the solution of the assembly process. Consequently, no customized device as an HMD is used and no connection to another computer is required. The parts to be assembled are identified by object recognition techniques in a way that markers are not used as an aid to the parts identification process and environment preparation is not needed. Instead of markers, models are used to identify the parts to be assembled.

On a setup phase, the proposed system loads the models and the assembly sequence representation from files or from the internet, as pictured on Fig. 1a. Models are composed by a label to identify the assembly part and edges. The edges are described by a label and sets of points, representing the beginning and the end of an edge. For each assembly part model, features are extracted and stored on a database. The assembly sequence is composed of a list of tasks and each task contains the sequence order and a set of connections. Each connection is composed by a set of labels which identifies the edges to be connected in order to join the subassemblies. A representation derived from [31] is used to represent the assembly sequence. The AST Table, described in [31], contains all possible solutions to the assembly process, whereas in the proposed system only one assembly solution is represented per assembly process, thus there is no need to represent the assembly state on each step. Only the assembly tasks column

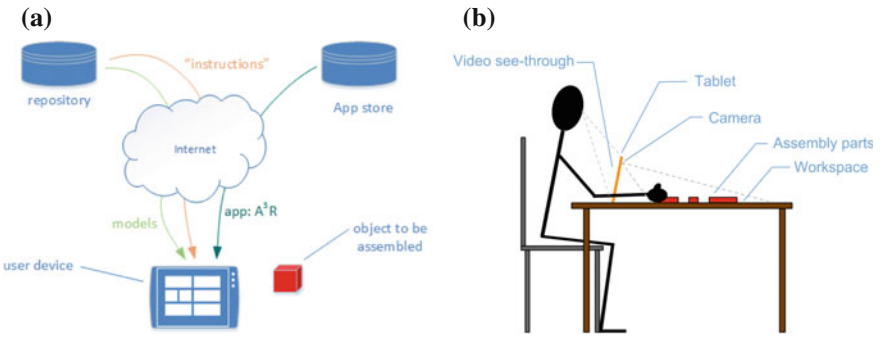

Fig. 1 Proposed system. **a** Models and instructions loading from the internet. **b** System setup

Table 2 Example of the proposed system assembly sequence representation

Sequence	Task
0	{I3}; {I0}
1	{T0, I2}
2	{W0, T6}; {W1, T5}

from the AST representation is used. An example of the assembly sequence representation is presented in Table 2. The Sequence column corresponds to the sequence order and the Task column represents the assembly task. An Assembly task contains sets of edges labels. On the Sequence 0, it is assumed that the first assembly part (I) is located at $(x, y) = (0, 0)$ on a cartesian base requiring only the edge that is on axis $x = 0$ and $y = 0$. Therefore, edges "I3" and"I0" are on $x = 0$ and $y = 0$ axis, respectively. On the Sequence 1, the Task column contains the connection between edge "T0" from the "T" assembly part and the edge "I2", from the "I" assembly part. Finally, on the Sequence 2, the Task column contains the connection between edge "W0" and "T6" and the connection between edges "W1" and "T5", from assembly parts "W" and "T", respectively. The assembly guidance is carried out by reading the table from the Sequence 0 to the last Sequence, stepping to the next Sequence whenever the connection between the subassemblies is detected.

Still on the setup phase, the intrinsic camera parameters and lens distortion are determined by a camera calibration process and stored on the database. Later, this information is used for the estimation of the object position and to correct the distortions caused by the lens on the camera image.

On the assembly guidance process phase, the tablet device is placed between the user and the assembly parts in a way that the user can see the assembly parts on the tablet's display in a video see-through display approach, as shown schematically on Fig. 1b.

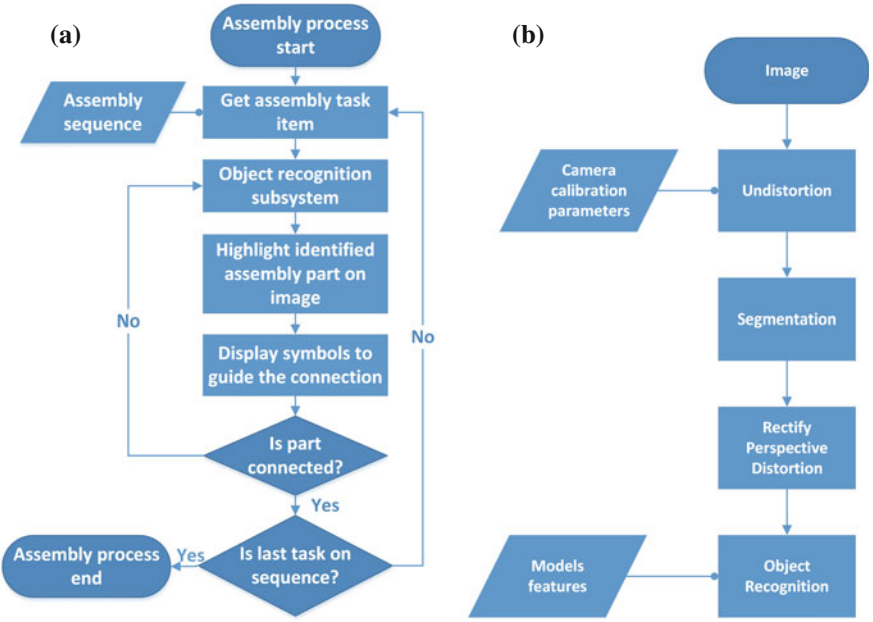

Fig. 2 System block diagram. **a** Assembly process diagram. **b** Object recognition subsystem

Figure 2a shows the assembly process algorithm. At first, the system retrieves an assembly task from the assembly sequence ordered list. If the assembly task is not the last one, the part associated with the assembly task is identified by the object recognition subsystem and is highlighted on the tablet's display to show to the user the part to move. Graphical signs are shown to the user indicating the following actions: flip the assembly part if necessary and connect the part to a subassembly. While a highlighted assembly part is not connected to the subassembly, the system continuously indicates to the user the assembly part and its placement for connection even while the user moves the part around the workspace, except in the case of occlusion that is not being taken care at this time. After the part is placed on the indicated location and the connection to the subassembly is detected, the system steps to the next assembly task. This loop is repeated until all parts are placed on assembly layout and the proper connections are validated, thus the assembly process is completed.

For each image frame, a sequence of image processing algorithms are executed on the object recognition subsystem, as shown on Fig. 2b. The first algorithm is executed to correct the lens distortion from the camera image by the use of the distortion lens parameters previously estimated on the setup phase. After the undistortion, objects on the workspace are segmented. Due to the tablet's camera position, segmented parts are rectified to remove the perspective distortion. Features are extracted from the segmented contours and then invariant features relative to rotation and scale are computed. On the object recognition algorithm, features are used to match the parts and the models.

4 Implementation

The implemented system uses an iPad Air 2 as platform with a video see-through approach. An 18 mm focal distance wide angle lens was placed on the tablet camera to broaden the field of view. Xcode was used to implement the software and the OpenCV library was used to develop the image processing algorithms. All implemented routines were written with Objective-C and C++ languages.

As a use case, the system was implemented to guide the assembly process of a pentomino puzzle [32]. This puzzle is composed of 12 pieces where each piece is the result of 5 square concatenated by their edges. Figure 3 shows the pentomino puzzle pieces and the given name of each piece that is used to identify each individual part and in the assembly sequence table. A common puzzle objective is to tile all the pieces in a rectangular box with an area of 6×10 squares. This particular configuration has 2,339 possible solutions. This application was selected because it represents a generic assembly process with known solutions where the user can place the pieces in well defined positions starting from a random distribution of pieces on the workspace.

On the setup phase, the camera calibration is performed using a method implemented in the OpenCV library [33]. Using the tablet's camera with a wide angle lens attached, twenty images were taken from a chessboard on arbitrary orientations to estimate the camera intrinsic matrix and lens distortion parameters. If the sum of the

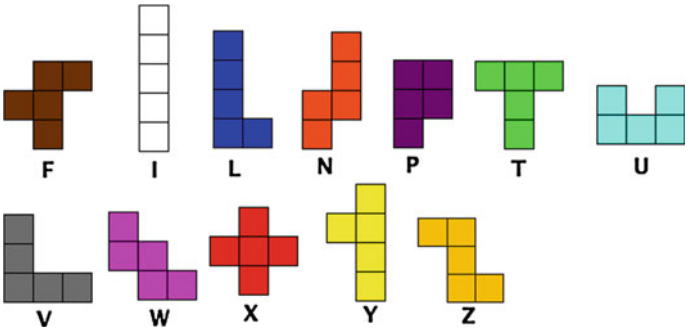

Fig. 3 Pentomino pieces identified by letters

squared distances between the observed projection points and the projected points (re-projection error) is less than 0.5, the camera intrinsic matrix and lens distortion parameters are stored for later use. If the estimated re-projection error is not less than 0.5, the calibration process is repeated. The correction (undistortion) of the camera lens distortion is made in two steps: computing the mapping of points in the distorted space to the undistorted space and interpolating the pixels on the undistorted image after the mapping. The mapping result is the same on all camera images with the same lens, therefore, for optimization proposes it is cached once computed and only the interpolation of pixels on the undistorted images is processed for each camera image.

Assembly parts images were segmented in four steps: (1) color segmentation is made by thresholding the pixels by hue, saturation and value; (2) a Canny Edge algorithm [34] is used to detect edges; (3) a 2×2 rectangular element size is used on a morphology closing process to close open borders on the detected edges; (4) a Border Following algorithm, as described by [35], is used to join the borders in vectors which represents the border points in a counter-clockwise order.

In the particular case of the 2D puzzle solution guidance, it is necessary to make the rectification of perspective distortion to compute assembly parts as a top–down view. Given that the pieces on puzzle are on the plane $Z = 0$, we can relate a point d on the image to a point D on the plane $Z = 0$:

$$sd = HD \tag{1}$$

where H is the homography matrix and s is an arbitrary scale.

The image points are computed as if the image was taken by a virtual camera placed on $C_v = (0, 0, h_v)_g$ on the global coordinate in O, as shown on Fig. 4. Using the (1) a point in the coordinate system O can be represented in the coordinate system C as:

$$s_c d_c = A \begin{bmatrix} -\cos(\theta) & 0 & h \cdot \frac{\sin^2(\theta)}{\cos(\theta)} \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & h \cdot \sin(\theta) \end{bmatrix} D = H_c D \tag{2}$$

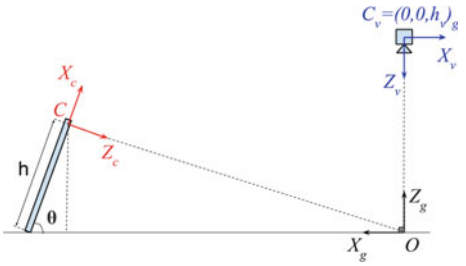

Fig. 4 Perspective effect removal: camera coordinate system C, global coordinate system O and virtual camera coordinate system V

and a point in the coordinate system O can be represented in the coordinate system V as:

$$s_v d_v = A \begin{bmatrix} \cos(\pi) & 0 & 0 \\ 0 & 1 & 0 \\ -\sin(\pi) & 0 & h_v \end{bmatrix} D = H_v D \tag{3}$$

where the d_c and d_v are the points in the images of the real camera and the virtual camera, respectively; A is the camera intrinsic matrix; D is the point on the plane of the assembly parts; H_x is the homography matrix of the real camera or virtual camera; θ is the device inclination angle; h is the distance between the camera and the device opposite side; h_v is the height of the virtual camera.

Relating the Eqs. (2) and (3), we can relate an image point to a point as taken by a virtual camera:

$$\alpha d_v = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & h_v \end{bmatrix} \begin{bmatrix} -\cos(\theta) & 0 & h \cdot \frac{\sin^2(\theta)}{\cos(\theta)} \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & h \cdot \sin(\theta) \end{bmatrix}_c^{-1} d_c \tag{4}$$

or

$$\alpha d_v = H_v H_c^{-1} d_c \tag{5}$$

where α is an arbitrary constant scale.

The distance h of the camera and the opposite side of the tablet is known and we used the gyroscope and accelerometer sensor of the tablet to compute the inclination angle θ.

Following the rectification, we used the Ramer–Douglas–Peucker algorithm [36, 37] implemented on OpenCV to approximate the segmented contours to polygons, which compressed the edge points and reduced noise. Precision of 1 % of contour length was used on the algorithm.

The approximated contours of the segmented parts change if the assembly part is rotated or if they are farther or near from the camera. The parts on the image are recognized by matching to the models of a database, so the extraction of features that

are invariant to translation, rotation and scale is necessary to make the comparison possible between segmented parts and a model in the database. The models in the database are loaded on the setup phase and they are described in more details later in this section. As presented in [38], the turning function is a description of a shape contour as a representation that is invariant to translation, rotation and scale. Suppose that a polygon is represented by a curve, then we can re-scale this curve in a way that its total length is equal to 1. The Turning Function $\phi(s)$ measures the change of the angle ϕ along the curve as a function of the curve length s. Following the curve's path to the counter-clockwise direction, the angle increases on counter-clockwise turns and decreases on clockwise turns, as the turns are accumulated along the curve. The turning function has the following features: it is translation invariant, scale invariant, the rotation is represented by a shift on accumulated turns and the choice of the initial starting point corresponds to a shift on the curve segment length.

The matching between two polygons is done by calculating a distance as discussed in [39]. The method used is the Polygonal Method. The choice of the initial starting point affects the turning functions, so it is necessary to compute the minimum distance from all possible turning functions computed by different initial starting points. The following metric is used to make the curve matching between two closed polygons A and B given their turning functions $\phi_1(s)$ and $\phi_2(s)$:

$$D(A,B) = \min_{\substack{\alpha\in\Re \\ u\in[0,1]}} \left[\int_0^1 (\phi_1(s) - \phi_2(s+u) + \alpha)^2 ds \right]^{\frac{1}{2}} \tag{6}$$

The optimal α is:

$$\alpha = \int_0^1 [\phi_1(s) - \phi_2(s)]\, ds - 2\pi u \tag{7}$$

where u is the choice of a starting point and α is the rotation difference between the two polygons.

The turning function is computed for each segmented object. Then we compute the minimum distance between the segmented object turning function and each of the model's turning function in the database by the use of Eq. (6). The minimum is reached when the best orientation between the curves and the starting point is selected and a matching is detected when the minimum distance to the model is less than a defined threshold of 0.8. Due to the nature of the pentomino pieces, some flipped pieces are not recognized as the same object because the turning function of a flipped piece is not the same depending on the piece's shape. So the turning function of the pentominoes pieces that presented distinct turning function on the flipped state were included. In this way some pentominoes pieces have two possible turning functions, as is the case of the "P" piece.

Each assembly part on the pentominoes puzzle has a unique shape, therefore it is necessary to treat repeated identified parts in the object recognition algorithm. If

a model is matched to more than one contour, the segmented image object with the minimum distance to the model is held as a match to the model. Yet, due to noise on the captured image, the segmentation algorithm can detect this noise as a small contour. Elimination of this noise contour is made by comparing the arc length of contours matched to the same model. If the arc length of one contour is less than 20 % of the comparing contour, the contour with the minor arc length is discarded as a possible match to the model.

For the assembly guidance process the implemented system loads the assembly parts models and the solution description from files on the setup phase. The assembly parts models are loaded by a JSON formatted file which lists the assembly parts with the following information:

1. Assembly part label;
2. Set of connected edges, each edge is labeled and described by vertices.

On Fig. 5 a JSON formatted file with the description of a pentomino part model is shown. In this example, the pentomino piece "V" is described by a label and by an array of edges. The edges are described by a label "Vn" (with n between 0 and 5) and the model's vertices. Each vertex is described by a point on a cartesian base. The models from the JSON formatted file are stored on a database and the model's turning functions for object recognition are derived from them.

Figure 6 shows a JSON formatted file with a description of an assembly sequence. The assembly sequence is represented by a list of assembly tasks containing:

1. "Sequence"—the sequence order;
2. "Part"—the associated assembly part model to be moved on the task and;
3. "Connections"—the assembly connections. The assembly connections are represented by a set of connected edges.

An assembly layout is used for guidance on user interface. The assembly layout is a grid of cells displayed on the camera image to guide the connection of assembly parts by indicating the part location on the whole object. The location of an assembly part is shown by highlighting the cells to be occupied, determined by the assembly task. Computing which cells are to be highlighted on the assembly task is made by:

1. reading the models and solution description;
2. translating and rotating the assembly part on each assembly task for the assembly part placement on assembly layout;
3. estimating the cells to be highlighted on each assembly task.

The assembly part model rotation is computed by rotating the associated part on the assembly task until all connected edges are parallel. Translation is computed by estimating the distance from the connected edges, which are all parallel after rotation. Given the rotation and translation of the assembly part model the validity of the connection is verified. A connection is valid whenever the rotated and translated part do not overlap another subassembly. The systems steps through each assembly task until all the parts are verified to have valid connections. When this condition occurs and it is in the end of the assembly sequence, the object is considered to be completely assembled.

```
{   "Parts":
       [{
          "Label": "V",
          "Edges":
             [{
                "Label": "V0",
                "Vertices": [{"X": 0,"Y": 0},{"X": 0,"Y": 300 }]
              },
              {
                "Label": "V1",
                "Vertices": [{"X": 0,"Y": 300},{"X": 100,"Y": 300 }]
              },
              {
                "Label": "V2",
                "Vertices": [{"X": 100,"Y": 300},{"X": 100,"Y": 100 }]
              },
              {
                "Label": "V3",
                "Vertices": [{"X": 100,"Y": 100},{"X": 300,"Y": 100 }]
              },
              {
                "Label": "V4",
                "Vertices": [{"X": 300,"Y": 100},{"X": 300,"Y": 0 }]
              },
              {
                "Label": "V5",
                "Vertices": [{"X": 300,"Y": 0},{"X": 0,"Y": 0 }]
              },
             ]
        },
        ...
       ]
}
```

Fig. 5 Assembly part JSON example. In this example the “V” part of the pentomino puzzle is described on Parts array

```
{
   "AssemblySequence":[
      ...,
      {
        "Sequence": "2",
        "Part": "L",
        "Connections": [{"Edges":["L4","W5"]},{"Edges":["L5","W4"]}]
      },
      ...
   ]
}
```

Fig. 6 Assembly sequence JSON example

5 Results and Discussion

We processed 35 images with the 12 pentominoes pieces displaced on random orientation and position without occlusion to evaluate the segmentation and object recognition algorithms. From a total of 420 pieces, the algorithm segmented 413 pieces, thus, the segmentation ratio is 0.98.

Analyzing the pieces which could not be segmented, we noticed that the segmentation failed due to borders that where not fully closed. The morphology closing operation with one interaction of a 2×2 rectangular element size was not sufficient to close some borders and the border following algorithm accumulated the points of the border as an open contour. The segmentation result may be improved if a larger element is used on the morphology closing operation. By the other hand, larger elements may deform the shape, which makes the identification task more challenging.

For the object recognition we assumed that the pentominoes pieces are unique. On the distance computation between segmented contours and models turning functions only segmented contours turning function with distance lower than 0.8 are selected as candidates for identification. As the object recognition depends directly from the segmentation result, we only used successfully segmented parts for the object recognition. Then, from a total of 413 segmented parts, 407 were correctly identified. Figure 7 demonstrates an example of the identified parts and their labels. We noticed that some of the unrecognized puzzle parts were near the edge of camera image. They were affected by lens distortion correction that acts more aggressively on pixels far from the center of image, which deforms the piece shape.

It is important to evaluate the execution time of each algorithm as near real time performance is desired on AR systems. We placed pentomino pieces in a way that the camera could capture all the 12 pieces and executed the algorithms 500 times. Execution time for each process is summarized on Table 3. Segmentation execution time had a great significance on the whole process execution time. As assembly guidance algorithm is executed in parallel on the implemented system, the assembly guidance algorithm execution time do not affect the image processing and the object

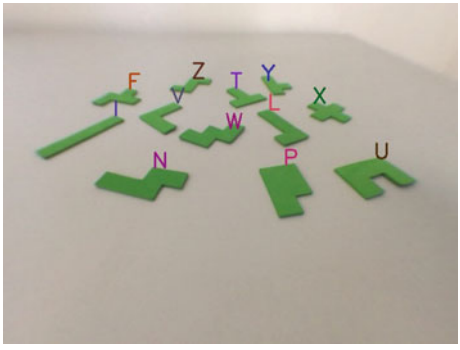

Fig. 7 Example of pentomino parts recognition assuming unique pentominoes

Table 3 Algorithms execution time

Algorithm	Time (ms)	
	Mean	Standard deviation
Undistortion	19.1	2.18
Segmentation	68.9	4.39
Perspective rectification	0.435	0.0706
Object recognition	20.7	1.71
Total	*109*	*8.35*

recognition execution time. Segmented contours turning function derivation and distance computation can be parallelized, therefore, the object recognition execution time can be improved by running part of the algorithm on a parallel process.

When an assembly part is required to be flipped, the system alternates between highlighting the current assembly part in white and green shapes signs, as shown in Fig. 8a, b, respectively, in order to indicate to the user the need for this action. The system produces an audible warning whenever the need for a flip action is detected. Figure 9 presents an intermediate state of the assembly guidance process. The assembly part to be moved on this sequence is highlighted and an arrow indicates to the user where the part must be moved to in order to connect to the subassembly. The highlighted cells near the arrow tip represent the placement location of the part for connection and the remaining filled cells represent already assembled and connected parts, or subassemblies, on previous steps. When the system detects that the part is placed on the right position and is connected to the subassembly, an audible confirmation is produced to inform the user and the system steps to the next assembly sequence. This process is repeated until the assembly process is detected as completed, as illustrated in Fig. 9b.

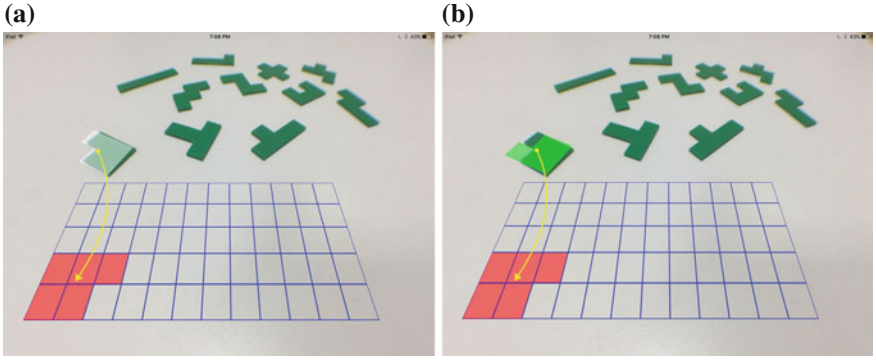

Fig. 8 Indication to the user of part to be flipped. **a** Alternating symbol for current detected part in *white*. **b** Alternating symbol for part in final flipped state in *green*

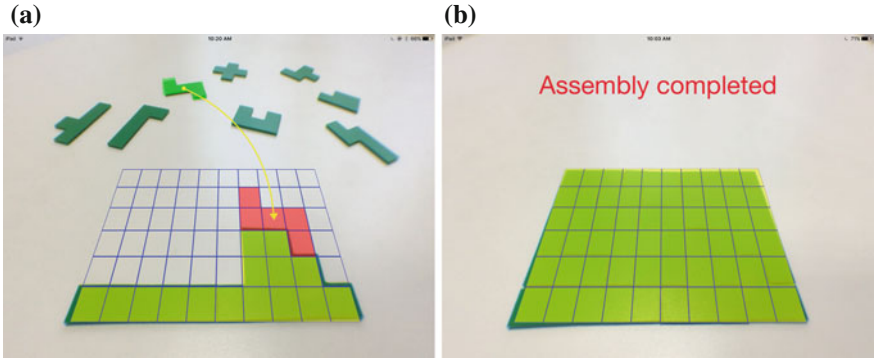

Fig. 9 Assembly process guidance viewed by the user. **a** Assembly process on an intermediate state. **b** Completed assembly process

6 Conclusion

This paper presented an AR system to guide an user through an assembly process. The system is based on image processing techniques to detect the parts to be assembled, to follow up the assembly process and uses graphical signs to inform the user of each step in the assembly sequence. Segmentation and object recognition are two important aspects of this project. Assembly parts segmentation is made by a combination of color segmentation, Canny Edge algorithm, morphology closing operation and border following algorithm. Segmentation rate of 0.98 has been achieved. Segmented assembly parts are identified by computing the turning function and the distance between the turning functions of segmented assembly parts and models. We made the assumption that all the 12 pentomino parts are present on the evaluated images and a filter algorithm have been used to reduce the likelihood of noise contour being incorrectly labeled as a pentomino part. We achieved a rate of 0.98 for object recognition with this process.

We evaluated the execution time of each image processing step: undistortion, perspective rectification, segmentation and object recognition. A mean of approximately 109 ms to the whole process execution time and an execution time of 20.7 ms for object recognition of 12 pentomino pieces were achieved in an iPad Air 2. We reckon that in about two generations of hardware upgrades real time performance will be possible.

The graphical interface guides the user through a series of movement operations such as flipping and connecting a part to a subassembly. Color symbols and audible confirmations and warnings give the necessary instructions and feedback to the user. The system has been implemented and tested with several puzzle solutions, from start point, where all parts are placed randomly in the workspace, until the full assembly

is completed. General descriptions of the assembly model, assembly task, assembly sequence and assembly layout were proposed, derived from [31], implemented and validated.

References

1. Tang, A., Owen, C., Biocca, F., Mou, W.: Comparative effectiveness of augmented reality in object assembly. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 73–80. ACM Press, New York, New York, USA (2003). doi:10.1145/642625.642626
2. Wiedenmaier, S., Oehme, O., Schmidt, L., Luczak, H.: Augmented reality (AR) for assembly processes design and experimental evaluation. Int. J. Hum. Comput. Interact. **16**(3), 497–514 (2003)
3. Baird, K.M., Barfield, W.: Evaluating the effectiveness of augmented reality displays for a manual assembly task. Virtual Real. **4**(4), 250–259 (1999). doi:10.1007/BF01421808
4. Caudell, T., Mizell, D.: Augmented reality: an application of heads-up display technology to manual manufacturing processes. In: Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences, pp. 659–669 vol. 2 (1992). doi:10.1109/HICSS.1992.183317
5. Feiner, S., Macintyre, B., Seligmann, D.: Knowledge-based augmented reality. Commun. ACM **36**(7), 53–62 (1993). doi:10.1145/159544.159587
6. Reiners, D., Stricker, D., Klinker, G., Müller, S.: Augmented reality for construction tasks: Doorlock assembly. In: Proceedings of the IEEE and ACM IWAR (1998)
7. Odenthal, B., Mayer, M.P., Kabuß, W., Schlick, C.M.: A comparative study of head-mounted and table-mounted augmented vision systems for assembly error detection. Hum. Factors Ergon. Manuf. Serv. Ind. **24**(1), 105–123 (2014). doi:10.1002/hfm
8. Boud, A., Haniff, D., Baber, C., Steiner, S.: Virtual reality and augmented reality as a training tool for assembly tasks. In: 1999 IEEE International Conference on Information Visualization (Cat. No. PR00210), pp. 32–36. IEEE Comput. Soc (1999). doi:10.1109/IV.1999.781532
9. Zauner, J., Haller, M., Brandl, A., Hartman, W.: Authoring of a mixed reality assembly instructor for hierarchical structures. In: Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003, pp. 237–246. IEEE Comput. Soc (2003). doi:10.1109/ISMAR.2003.1240707
10. Salonen, T., Sääski, J., Hakkarainen, M., Kannetis, T., Perakakis, M., Siltanen, S., Potamianos, A., Korkalo, O., Woodward, C.: Demonstration of assembly work using augmented reality. In: Proceedings of the 6th ACM International Conference on Image and Video Retrieval - CIVR '07, pp. 120–123. ACM Press, New York, New York, USA (2007). doi:10.1145/1282280.1282301
11. Henderson, S., Feiner, S.: Exploring the benefits of augmented reality documentation for maintenance and repair. IEEE Trans. Vis. Comput. Graph. **17**(10), 1355–1368 (2011)
12. Wang, Z.B., Ong, S.K., Nee, A.Y.C.: Augmented reality aided interactive manual assembly design. Int. J. Adv. Manuf. Technol. **69**(5–8), 1311–1321 (2013). doi:10.1007/s00170-013-5091-x
13. Khuong, B.M., Kiyokawa, K., Miller, A., La Viola, J.J., Mashita, T., Takemura, H.: The effectiveness of an AR-based context-aware assembly support system in object assembly. In: 2014 IEEE Virtual Reality (VR), pp. 57–62 (2014). doi:10.1109/VR.2014.6802051
14. Chiang, H.K., Chou, Y.Y., Chang, L.C., Huang, C.Y., Kuo, F.L., Chen, H.W.: An augmented reality learning space for PC DIY. In: Proceedings of the 2nd Augmented Human International Conference, p. 12. ACM Press, New York, New York, USA (2011). doi:10.1145/1959826.1959838

15. Kitagawa, M., Yamamoto, T.: 3D puzzle guidance in augmented reality environment using a 3D desk surface projection. In: 2011 IEEE Symposium on 3D User Interfaces (3DUI), pp. 133–134. IEEE (2011). doi:10.1109/3DUI.2011.5759241
16. Fiorentino, M., Uva, A.E., Gattullo, M., Debernardis, S., Monno, G.: Augmented reality on large screen for interactive maintenance instructions. Comput. Ind. **65**(2), 270–278 (2014). doi:10.1016/j.compind.2013.11.004
17. Hakkarainen, M., Woodward, C., Billinghurst, M.: Augmented assembly using a mobile phone. In: 2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality pp. 167–168 (2008). doi:10.1109/ISMAR.2008.4637349
18. Nakanishi, M., Ozeki, M., Akasaka, T., Okada, Y.: Human factor requirements for applying augmented reality to manuals in actual work situations. In: 2007 IEEE International Conference on Systems, Man and Cybernetics, pp. 2650–2655 (2007). doi:10.1109/ICSMC.2007.4413588
19. Li, Y., Kameda, Y., Ohta, Y.: AR replay in a small workspace. In: 2013 23rd International Conference on Artificial Reality and Telexistence (ICAT), pp. 97–101. IEEE (2013). doi:10.1109/ICAT.2013.6728913
20. Murakami, K., Kiyama, R., Narumi, T., Tanikawa, T., Hirose, M.: Poster: A wearable augmented reality system with haptic feedback and its performance in virtual assembly tasks. In: 2013 IEEE Symposium on 3D User Interfaces (3DUI), pp. 161–162. IEEE (2013). doi:10.1109/3DUI.2013.6550228
21. Yuan, M.L., Ong, S.K., Nee, A.Y.C.: Augmented reality for assembly guidance using a virtual interactive tool. Int. J. Prod. Res. **46**(7), 1745–1767 (2008). doi:10.1080/00207540600972935
22. Novak-Marcincin, J., Barna, J., Janak, M., Novakova-Marcincinova, L., Torok, J.: Visualization of intelligent assembling process by augmented reality tools application. In: 2012 4th IEEE International Symposium on Logistics and Industrial Informatics, pp. 33–36. IEEE (2012). doi:10.1109/LINDI.2012.6319505
23. Serván, J., Mas, F., Menéndez, J., Ríos, J.: Assembly work instruction deployment using augmented reality. Key Eng. Mater. **502**, 25–30 (2012). doi:10.4028/www.scientific.net/KEM.502.25
24. Alvarez, H., Aguinaga, I., Borro, D.: Providing guidance for maintenance operations using automatic markerless augmented Reality system. In: 2011 10th IEEE International Symposium on Mixed and Augmented Reality pp. 181–190 (2011). doi:10.1109/ISMAR.2011.6092385
25. Wang, Z.B., Shen, Y., Ong, S.K., Nee, A.Y.C.: Assembly design and evaluation based on bare-hand interaction in an augmented reality environment. In: 2009 International Conference on CyberWorlds, pp. 21–28. IEEE (2009). doi:10.1109/CW.2009.15
26. Ong, S., Wang, Z.: Augmented assembly technologies based on 3D bare-hand interaction. CIRP Ann. - Manuf. Technol. **60**(1), 1–4 (2011). doi:10.1016/j.cirp.2011.03.001
27. Sääski, J., Salonen, T., Hakkarainen, M., Siltanen, S., Woodward, C., Lempiäinen, J.: Integration of design and assembly using augmented reality. Micro-Assem. Technol. Appl. **260**, 395–404 (2008). doi:10.1007/978-0-387-77405-3_39
28. Westerfield, G.: Intelligent Augmented Reality Training for Assembly and Maintenance. Master of science, University of Canterbury (2012)
29. Carmigniani, J., Furht, B., Anisetti, M., Ceravolo, P., Damiani, E., Ivkovic, M.: Augmented reality technologies, systems and applications. Multimed. Tools Appl. **51**(1), 341–377 (2010). doi:10.1007/s11042-010-0660-6
30. Homem de Mello, L.S., Sanderson, A.C.: Representations of mechanical assembly sequences. IEEE Trans. Robot. Autom. **7**(2), 211–227 (1991). doi:10.1109/70.75904
31. Gottipolu, R.B., Ghosh, K.: A simplified and efficient representation for evaluation and selection of assembly sequences. Comput. Ind. **50**(3), 251–264 (2003). doi:10.1016/S0166-3615(03)00015-0
32. Golomb, S.W.: Polyominoes: puzzles, patterns, problems, and packings, 2nd edn. Princeton University Press, Princeton (1996)
33. Zhang, Z.: Flexible camera calibration by viewing a plane from unknown orientations. In: Proceedings of the Seventh IEEE International Conference on Computer Vision **1**(c), 666–673 (1999). doi:10.1109/ICCV.1999.791289

34. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
35. Suzuki, S., Abe, K.: Topological structural analysis of digitized binary images by border following. Comput. Vis., Graph., Image Process. **30**(1), 32–46 (1985). doi:10.1016/0734-189X(85)90016-7
36. Ramer, U.: An iterative procedure for the polygonal approximation of plane curves (1972). doi:10.1016/S0146-664X(72)80017-0
37. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartogr.: Int. J. Geogr. Inf. Geovis. **10**(2), 112–122 (1973). doi:10.3138/FM57-6770-U75U-7727
38. Arkin, E.M., Chew, L.P., Huttenlocher, D.P., Kedem, K., Mitchell, J.S.B.: An efficiently computable metric for comparing polygonal shapes. IEEE Trans. Pattern Anal. Mach. Intell. **13**(3), 209–216 (1991). doi:10.1109/34.75509
39. Cakmakov, D., Celakoska, E.: Estimation of curve similarity using turning functions. Int. J. Appl. Math. **15**, 403–416 (2004)