CEIG - Spanish Computer Graphics Conference (2016)
Alejandro García-Alonso and Belen Masia (Editors)

Augmented Reality system to assist in manufacturing processes

R. J. Ugarte[1], N. Barrena[1], H. V. Diez[1], H. Alvarez[1] and D. Oyarzun[1]

[1]Interactive Computer Graphics Department, Vicomtech-IK4
20009 Donostia-San Sebastian, Spain

Abstract

This paper presents the initial results of a research project focused on bringing the Markerless Augmented Reality technology to the advance manufacturing sector. These initial results consist in a completely functional software system where a very simple use case is implemented to validate a set of base technologies. It has been implemented using a conventional camera without placing any markers to not modify the working environment and to not interfere with other industrial systems. Moreover, stability is a strong requirement to obtain a system that is actually useful in a manufacturing environment. Therefore, it is expected that an advanced version of this technology can be applied in an actual industrial use case in a short-to-medium term.

Categories and Subject Descriptors (according to ACM CCS): I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Tracking

1. Introduction

In the very latest years, several international initiatives are promoting what is called the Fourth Industrial Revolution. Like the *Industrie 4.0* in Germany [KWH13], *Industrial Internet* in USA [EA13], *Industrial Value Chain Initiative* in Japan [bwo15], etc. These are roadmaps motivated by Governments, as country strategies, to make this concept real. This revolution is supposed to be driven by the application of different aspects and techniques over the actual factory, such as interoperable communication protocols, digitalization, advanced HCI techniques, etc. In this trend, it is considered that Visual Computing technologies should have a very important role.

Some authors carried out a recent research reflecting the usage of Visual Computing fields as Key Enabling Technologies -KET- for the Fourth Industrial Revolution [aotp15]. The work presented in this paper is a step forward, where a proof of concept for assisting manufacturing processes has been developed.

The proof of concept starts from the hypothesis that in the context of the new industrial trends, the advances on Human Machine Interfaces will play a key role, creating tools that improve the efficiency and competitiveness of workers using them.

The software system has been developed trying to fulfil these requirements:

- **To not use any marker.** The base technology is pure markerless. Therefore, flexibility is provided to use in any machine or process.
- **To place any kind of media content.** The tracking system should be robust enough to give support to any kind of media that is useful to assist in the manufacturing process.
- **To not interfere with other industrial software or hardware systems.** The final software and interface should be independent from any other industrial software, hardware or process. It will not be part of the critical steps in the production chain.
- **Importance of the stability.** The stability of the tracking will be crucial. It will be robust enough to provide support in processes that involve complex machines and granular operations.

Fulfilling these issues, the base system and the proof of concept to validate it have been developed as described in the rest of the paper.

The paper is structured as follows. The next section presents the state of the art on augmented reality technologies(from now on AR) specifically applied in the industrial sector. Section 3 details the technology involving our system and in Section 4 some experimental results are described. Finally, Section 5 presents the conclusions obtained and future work.

2. Related Work

Many authors have written about Markerless AR in the recent years. Good examples are provided by [TdMLA*07] or [CMPC06] where several markerless techniques are described in a general form, giving the initial clues for the developing of an AR markerless system that could be applied in many fields of knowledge.

Specifically, in the industry sector [NOCM12] makes a survey of AR systems and techniques that can be applied for several tasks

DOI: 10.2312/ceig.20161317

such as robot path planning, AR collaborative design, plant layout or simulation. They also present the main challenges for the future AR systems on the industrial scope, focusing on the human interaction with the environment.

The present trend of manufacturing focuses on the modern ways of human interaction with the workstation in order to help on common tasks giving extra information about the workstation and the manufacture.

Given that, there are many existing solutions to reach that. For example, AR solutions for industrial purposes based on markers, such as the system proposed by [HWB08]. Here an AR system for mobile phones is presented, where the 3D virtual model is renderized in a position given by the detected marker.

Some authors have oriented their systems to several methods based on interaction in the manufacturing area such as [KVPK15], where the human-robot interaction is used to give the worker a real-time visualization of the robot's plans. In this case they place some artificial markers into the environment and extract features from it to recreate a 3D virtual representation, so that in the real-time stage, the markers are not necessary.

A similar strategy is used by [BPS05], where they place markers on the reconstruction stage and manually calibrate the system to place the CAD model on the environment correctly.

The main disadvantage of those methods is that you need to place the markers along the environment and in many cases it is not possible. Markerless AR techniques have evolved lately trying to overcome that situation.

For example, in [UWS09] a CAD-based recognition system is presented, where a 3D model can be directly tracked without the use of markers.

Moreover, [WLT*16] describes an AR markerless system, without the need of using markers on the reconstruction stage, that guides a worker through an assembly manual giving instructions with 3D virtual models placed directly on the workstation. The advantage with the previous system is that this solution can reconstruct itself the environment and does not need a CAD model to work. The main disadvantage is the need of an expensive RGB-D camera.

The reason that gives rise to this paper is the possibility to create a complete system using a conventional camera to create a 3D reconstruction of the environment and track it without placing any artificial markers. This paper presents a completely functional, simple to use and low-cost Markerless AR system for an interactive guiding of a worker on the assembly tasks.

3. System Overview

The main goal of this work is to create the base technology to develop tools that assist on manufacturing processes.

As mentioned previously, one key requirement is to not interfere in the production processes. Therefore, the system is conceptually defined as it is shown in figure 1, where the own production processes and machinery are separated from the assisted manufacturing processes.

In order to properly use AR techniques with industry or fabrication purposes it is necessary to set up correctly the virtual elements in the nearest environment, in this case the workstation. With this objective, the main goal of the system described in this section is to estimate accurately the camera pose in real-time. This pose must be calculated from the input frames that the camera is providing.

The aim of the design of the system presented in this work is to not pollute the environment with external marks. This way, the tracking system is based in the 3D representation of the nearest environment in order to estimate the camera pose, avoiding the use of marker based techniques.

In a synthesized way (figure 3), the system behaviour is as follows: it takes into account a set of images, which represent the nearest environment and it uses Structure from Motion (from now on SfM) techniques to obtain a 3D representation.

Therefore, the 3D representation of the environment is obtained, as well as the camera poses for all the images of the set used in the SfM process. Later, the camera pose will be obtained for each input frame and for this purpose it will be necessary to match the points of the 3D representation with their corresponding 2D points in the current images, as well as apply a template matching techniques to improve the pose estimation of the points.

In order to make the described process accurate, the system presented in this work follows two main stages:

- **Preprocess Stage**
  This stage is performed off-line and the main goal is to get the 3D representation of the environment. This is done using SfM techniques. A set of images of the nearest environment, denominated *keyFrames* are used as the input of this process. These *keyFrames* are used in the SfM process in order to obtain the 3D representation of that nearest environment, i.e the 3D point cloud (sample can be seen in figure 2). For this reason, the set of images must cover different perspectives of the environment (or object) where the AR application is going to work.
- **Real-time Stage**
  This stage is launched online and the *keyFrames* and the 3D point cloud obtained in the preprocess stage is used. For each input frame, the main goal is to calculate the camera pose. Two steps define this stage: detection and tracking. The first step is done matching the *keyPoints* (the most characteristic 2D points of the image) with their corresponding 3D points of the point cloud, obtaining an initially camera pose. In the second step template matching techniques are used in order to track the points of the point cloud through the previous input image to the current one, again obtaining the camera pose. When the tracking of the points is lost the system returns to the first step (figure 3).

The following sections are devoted to describe in detail the two stages of the described algorithm.

3.1. Preprocess Stage

The goal of the Preprocess Stage is to create a 3D representation of the environment from a given set of images. This representation needs to be as accurate as possible to avoid errors in the Real-time Stage. SfM techniques are used to achieve this purpose.

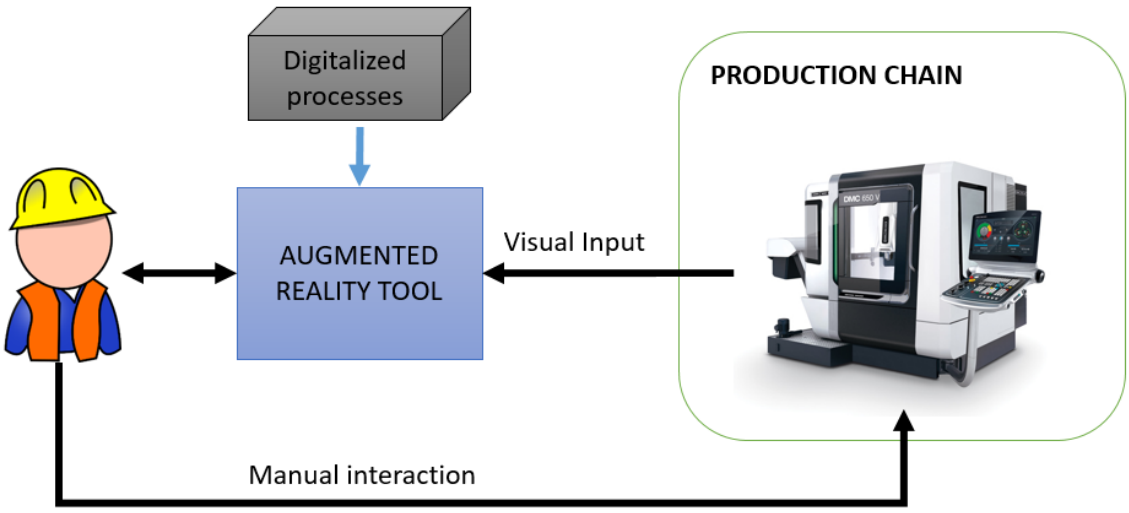

Figure 1: *Schematic of an industrial AR system*

These techniques use *keyFrames* as input data. For each *keyFrame*, *keyPoints* are extracted and theirs descriptors are calculated. In this way, FAST [TH98] feature extractor is used in order to extract the *keyPoints* of the *keyFrame*.

On the other way, the system has been built specifically for FREAK descriptors [AOV12] in order to work in real-time. This kind of descriptor is based on the human vision pattern. FREAK are binary descriptors calculated by the difference of intensity between the points of the pattern. Due to that it is very robust against illuminance and orientation changes but it is not scale invariant. To solve that issue, scale pyramidal reductions of the *keyFrames* are computed. For every scale level the *keypoints* are extracted. And for every *keypoint* its descriptor is calculated applying levels of Gaussian noise, taking the mean descriptor of the Gaussian levels as the descriptor of the *keypoint*.

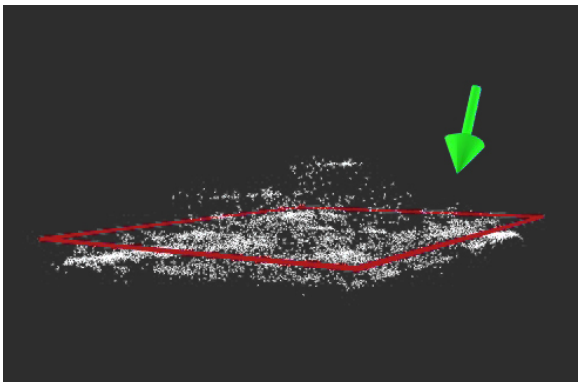

Figure 2: *Side view of the 3D point cloud*

Once the descriptors are calculated, in a sequential way, the algorithm tries to match those descriptors of the current *keyFrame* with the previous ones. In order to apply the SfM process as mentioned before, the same *keyPoint* had to be seen in several *keyFrames*. So, with the 2D positions of the same *keyPoint* in the different *keyFrames*, a triangulation method is applied in order to get their 3D coordinates. Then, a Bundle Adjustment [TMHF99] process is launched to get an estimation of the *keyPoints* 3D coordinates. Also a normal vector and a *reference keyFrame* has to be assigned for each *keyPoint*. In this case, the *reference keyFrame* for each *keyPoint* is the one where the *keyPoint* had been seen firstly and the normal vector associated to it faced the camera, i.e. is a vector from the *keyPoint* to the camera center.

The output of the Preprocess Stage is a file with the following data:

1. Point cloud (3D coordinates of the reconstructed points).
2. Surface normal estimation of each point of the point cloud.
3. *Reference keyFrame* for each point of the point cloud,
4. Camera pose of each *keyFrame* and,
5. Measurements of each point of the point cloud in each *keyFrame*.

It is important to remark that in order to simplify, the normal vector estimation for each *keyPoint* faces its *reference keyFrame* . Those normal vectors will be used later in the tracking stage.

Once the point cloud is calculated, all the *keyFrames* are processed to prepare them for the Real-time stage. For every *keyFrame* two actions are performed.

First, a 3D process in which a three level pyramidal reduction of the *keyFrame* is built. For each level, the *keyFrame* and its *keyPoints* from the 3D point cloud are resized to a half of the previous size.

Furthermore, each level is smoothed with a Gaussian kernel. The Gaussian smooth is applied to simulate intermediate scale levels

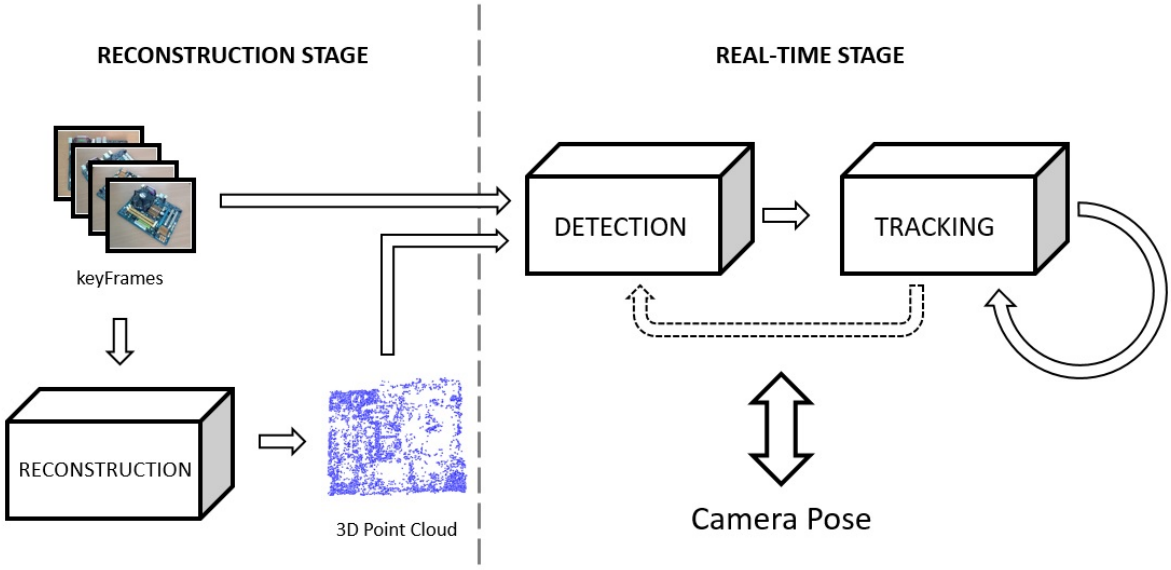

Figure 3: *Conceptual view of the system architecture*

and the effects of blur or noise during the live capture. The mean value obtained from the descriptors extracted in each smoothed image represents robustly the *keyPoint* and is the one that is saved as the representative descriptor of the *keyPoint*.

Second, a 2D process of the image where independent *keyPoints* of the *keyFrames*, not related with the 3D point cloud, are extracted. This *keyPoints* are used later for a refinement of the pose during the real-time stage and for the construction of a global database of all the *keyFrames*, using a fixed number of *keyPoints* for every *keyFrame*. This 2D process also applies Gaussian smooth to compensate blur, scale and noise during the capture and the mean value obtained by computing the descriptors of all the Gaussian levels is saved. These *keyPoints* are filtered by similarity of their descriptors and a score is given to each one of them. The ones with less score are discarded to reduce the number of *keyPoints* in the database.

3.2. Real-time stage

Once the pre-process is done, there is a middle stage needed to track correctly the 3D point cloud generated in the previous stage. The system can not associate automatically the 3D point cloud to the 3D virtual model, to be displayed and to match it with the 3D point cloud. So, an authoring tool is implemented to scale, rotate and translate the virtual models to match with the 3D point cloud. Once the transformation is complete, the real-time stage begins (figure 3).

The goal of this stage is to estimate accurately the camera pose for every input frame. This stage is divided in two sub-stages, the detection mode, where the system gives a first camera pose and the tracking mode, that estimates the rest of the camera poses until it fails and goes again into detection mode, repeating the process (figure 3).

The input needed in this stage are the *keyFrames* used in the preprocess stage and all the information calculated on the preprocess stage about the *keyFrames* and the point cloud. Once the database ready and the adjustment of the virtual model is done, the algorithm starts automatically on detection mode.

3.2.1. Detection mode

This stage uses the information extracted from the 2D process of the preprocess stage. With the input camera frame, the algorithm extracts the *keyPoints* and computes the descriptors. After that, it tries to match them with the global database of all the *keyFrames*.

Within this step, the algorithm obtains a histogram of the number of descriptors that are matched correctly for every *keyFrame*. This histogram shows the similarity of the input frame with the collection of the *keyFrames*. In this case, in order to gain robustness, the three *keyFrames* most similar to the input frame are taken. This helps the system to match correctly more *keyPoints* of the input frame with the *keyPoints* of the 3D point cloud.

With the three *keyFrames*, a function matches the descriptors stored with the input ones. The output of the function is the number of *keyPoints* correctly matched (from now on inliers) and a homography between the input frame and the *keyFrame*. This homography represents the difference of perspective between them.

If the number of inliers is higher than a fixed number, the system goes into the refinement step. Here, for the three selected *keyFrames*, the system takes initial homographies and warps the input frame with them. This process tries to adapt the view of the input frame to the *keyFrames* perspective.

On the new warped input frame, the system also warps the points of the 3D point cloud and recalculates their descriptors. Then, those are matched with the input descriptors. The inliers that match

correctly are pushed into a vector of all the inliers of the three *keyFrames*.

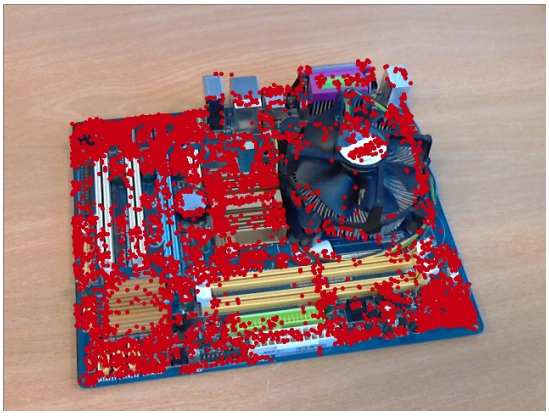

Figure 4: *Point cloud projected with the final camera pose*

Finally, the system will try to find the camera pose based on those inliers. For that, a *PnP* algorithm is applied, where the outliers are discarded using a *RANSAC* style procedure. If the final number of inliers is higher than a fixed minimum number, the final camera pose is taken as the correct one (figure 4), jumping to tracking mode.

3.2.2. Tracking Mode

The goal of this stage is to update the camera pose between the consecutive input frames. It is supposed not to be big shift differences between consecutive frames, so the 3D point cloud will be tracked following the points from one frame to the next.

To achieve that, based on the proximity of the camera pose between the previous frame and all the *keyFrames*, the algorithm uses the camera pose of the previous frame to find the current nearest *keyFrame*.

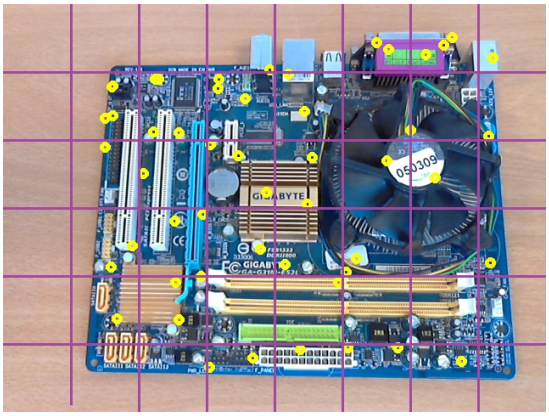

Figure 5: *Selected keyPoints taken from the 8x6 grid*

After that, the system selects a fixed number of *keyPoints* of the 3D point cloud that are visible in the selected *keyFrame*. With the aim of making the tracking properly the points are taken uniformly distributed from the 3D point cloud using a grid. This is done using a 2D grid with the projections of the 3D point cloud with the camera pose of the previous frame.

Figure 6: *Patch warped*

In this case we take 50 points using a 8x6 grid with 80x80 pixels square slots (figure 5). Jumping iteratively through each slot, the system takes the point with the best detectability and jumps to the next. If a slot does not have more points on it, the system jumps to the next one until it gets the 50 points or the maximum visible points.

Then, using template matching techniques, fixed size patches are taken around the selected projected points in the actual *keyFrame*(figure 5). More precisely, the patch size is 32x32 pixels being its center the projection of the 3D point. The corners of the patch are un-projected on the plane obtained with the normal vector associated to that 3D point and then projected to the plane of the current frame (figure 7), obtaining a warped square (figure 6). Then, the rest of the points of the patch are interpolated using the new coordinates of the corners.

Using a fixed searching area of 128x128 pixels, in our experiment, the patch and the searching area suffer a three level scale reduction in order to reduce computation complexity (figure 8).

In an iterative process, beginning with the smallest level and using a cross-correlation coefficient, the smallest reduction of the patch (8x8 pixels) is searched in the smallest level of the search area (32x32 pixels). Then, the position is propagated to the higher levels and finally to the original image, where the final position of the point is located.

Finally, the transferred points are used to feed the *PnP* algorithm that is executed in a *RANSAC* style. If the inliers are more than a fixed number, the resultant camera pose is taken as the new one. In other case, the system fails and the algorithm goes again to detection mode.

4. Experimental Results

This section evaluates the main characteristics of the system, such as, the robustness with regard of lighting or scale variations, partial occlusions, big image shift or the real-time requirement of the algorithm. All the code has been implemented in C++ and the PC used for the test is a Windows 10 with an Intel Core i5 3.3GHz 8GB.

The system has been tested with a CPU board as 3D real object.

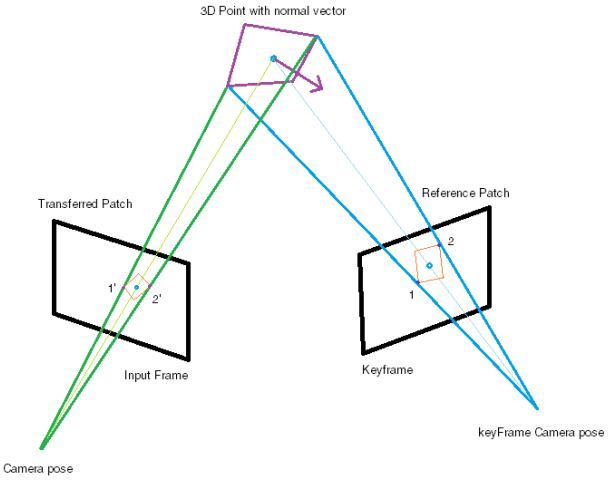

Figure 7: *Transferring patch process*

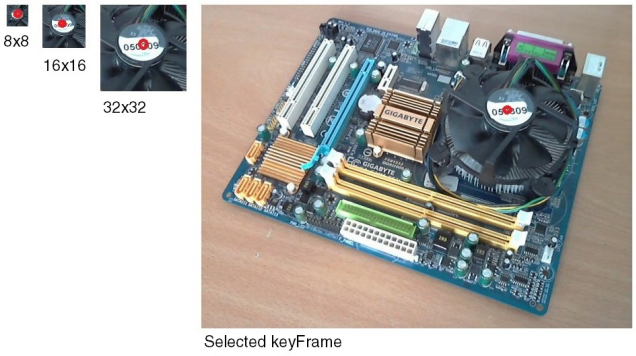

Figure 8: *Scale reductions of the patch*

The camera used for the test is a simple web camera with a VGA resolution (640x480 pixels). Twenty one photos of the board have been taken trying to cover the 360 degrees around the model and about 45 degrees from the aerial view.

In the real-time stage, the system takes a few minutes to compute the 3D point cloud and to process the *keyFrames* with that information. The 3D point cloud obtained once the images are passed through the reconstruction stage is a dense cloud with 6400 detectable 3D points.

In the real-time stage the detection mode, depending on the features of the input frame, runs approximately between 5 to 12 frames per second. This frame rate is a little below from real-time application but once the environment is detected, on the tracking mode, the system can run from 40 to 110 frames per second.

In the detection mode, to find the nearest *keyFrames* to the input frame, the global matcher takes 500 keypoints of each *keyFrame*, creating in this case a database of 4200 points. The bigger this database is, the slower the detection phase will run. After matching the input frame with the database, the experiments shows that no more than three *keyFrames* are needed to be extracted to perform the finest matching process. To reduce the compute time, this finest match is done parallelizing the three matching process. Once the first camera pose is obtained, the system jumps to tracking mode.

During the tracking mode, the system is continuously searching the *keyFrame* that is the closest to the input frame. It has been proved in the experiments, that in the process of transferring the warped patch, the complete tracking process is about two times faster if only the corners of the patch are transferred, while the rest of the points that belong to the patch are interpolated.

The system supports partial occlusions (figure 10), about 50% of occlusion, in both modes, detection and tracking. Also, in detection mode the system can make a correct match without the need of see all model. Furthermore the system can keep the tracking mode with heavy camera moves and can works in poor lighting conditions.

Finally, the result of the system is a stable tracking of the board where virtual objects can be placed (figure 9)

5. Conclusions and future work

This paper has described the initial results of a running research project whose objective is the development of an innovative technology that assists workers during manufacturing processes.

The partial results have consisted in the design and implementation of the base technology and a simple use case to validate it.

The system has been developed taking into account four critical requirements:

- To use pure markerless technology.
- To provide integration with any kind of media.
- To not interfere with industrial critical processes.
- To give enough stability.

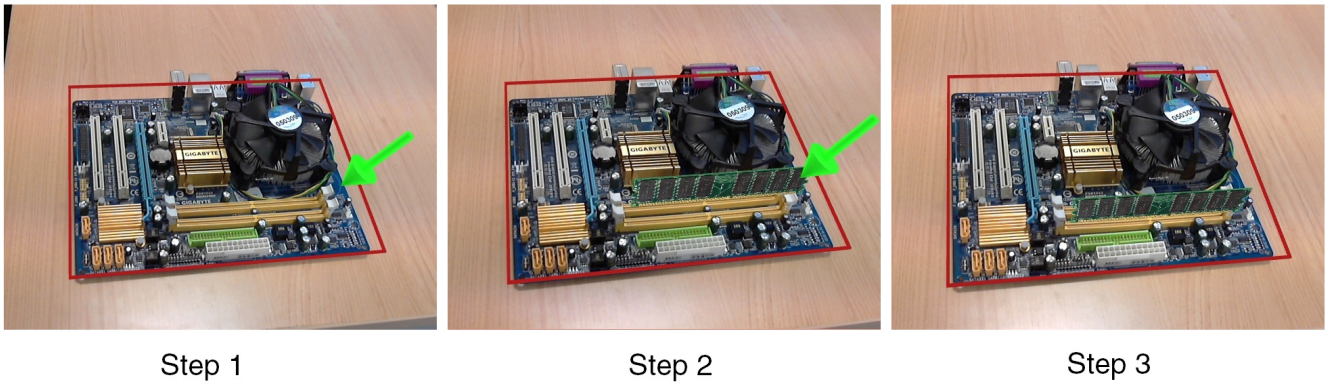

Figure 9: *Final application of the system; guiding on the assembly of a RAM memory piece by steps*

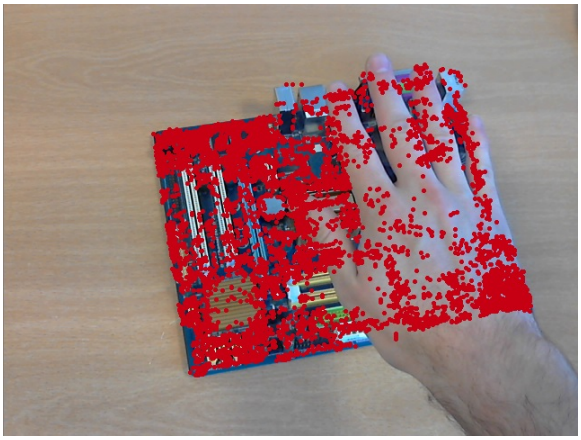

Figure 10: *Tracking mode with occlusion*

The results show the robustness of the system and they give sense to continuing the research towards a deployment in a real industrial use case.

Future work is focused on three main lines from the technological point of view:

- New tracking modules. A module that adds border tracking fed from machine CAD models is being implemented. This will strongly improve the tracking in cases where the CAD model exists.
- Optimization. The system is being optimized for its use in mobile devices, avoiding the need of having a PC during the assistance process.
- Web port. The tracker is being ported to web-based platforms.

In parallel, a real industrial use case is being developed and it will be validated in manufacturing companies with a closed relation to the author's research centre.

References

[aotp15] AUTHORS OF THIS PAPER S.: Visual computing as a key enabling technology for industrie 4.0 and industrial internet. *Computer Graphics and Applications, IEEE 35*, 2 (2015), 26–40. 1

[AOV12] ALAHI A., ORTIZ R., VANDERGHEYNST P.: Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (2012), Ieee, pp. 510–517. 3

[BPS05] BLESER G., PASTARMOV Y., STRICKER D.: Real-time 3d camera tracking for industrial augmented reality applications. 2

[bwo15] Industrial value chain initiative. "http://www.iv-i.org/", 2015. "[Online; accessed 20-February-2016]". 1

[CMPC06] COMPORT A. I., MARCHAND E., PRESSIGOUT M., CHAUMETTE F.: Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *Visualization and Computer Graphics, IEEE Transactions on 12*, 4 (2006), 615–628. 1

[EA13] EVANS P. C., ANNUNZIATA M.: Industrial internet: Pushing the boundaries of minds and machines. *General Electric Co. Imagination at work* (2013). 1

[HWB08] HAKKARAINEN M., WOODWARD C., BILLINGHURST M.: Augmented assembly using a mobile phone. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on* (2008), IEEE, pp. 167–168. 2

[KVPK15] KOŠNAR K., VICK A., PŘEUČIL L., KRÜGER J.: Markerless augmented reality for human robot interaction. In *Modelling and Simulation for Autonomous Systems*. Springer, 2015, pp. 185–195. 2

[KWH13] KAGERMANN H., WAHLSTER W., HELBIG J.: Recommendations for implementing the strategic initiative industrie 4.0. *Plattform Industrie 4.0* (2013). 1

[NOCM12] NEE A., ONG S., CHRYSSOLOURIS G., MOURTZIS D.: Augmented reality applications in design and manufacturing. *CIRP Annals-Manufacturing Technology 61*, 2 (2012), 657–679. 1

[TdMLA*07] TEICHRIEB V., DO MONTE LIMA J. P. S., APOLINÁRIO E. L., DE FARIAS T. S. M. C., BUENO M. A. S., KELNER J., SANTOS I. H.: A survey of online monocular markerless augmented reality. *International Journal of Modeling and Simulation for the Petroleum Industry 1*, 1 (2007). 1

[TH98] TRAJKOVIĆ M., HEDLEY M.: Fast corner detection. *Image and vision computing 16*, 2 (1998), 75–87. 3

[TMHF99] TRIGGS B., MCLAUCHLAN P. F., HARTLEY R. I., FITZGIBBON A. W.: Bundle adjustmentâĂŤa modern synthesis. In *Vision algorithms: theory and practice*. Springer, 1999, pp. 298–372. 3

[UWS09] ULRICH M., WIEDEMANN C., STEGER C.: Cad-based recognition of 3d objects in monocular images. In *ICRA* (2009), vol. 9, pp. 1191–1198. 2

[WLT*16] WU L.-C., LIN I., TSAI M.-H., ET AL.: Augmented reality instruction for object assembly based on markerless tracking. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2016), ACM, pp. 95–102. 2