

# An Interactive Vision-based 3D Reconstruction Workflow for Industrial AR Applications

Marion Langer\*  
metaio GmbH

Selim BenHimane†  
metaio GmbH

## ABSTRACT

“Knowledge about the real environment allows the interaction between virtual and real objects in almost all Augmented Reality (AR) scenarios. It increases the number of possible tasks that can be faced with AR and it improves the way how AR is experienced.” These statements are well known by the AR research community. However, as AR itself, they took some time to win the trust of industrial users which are targeted by many AR developers. It took so long because the technology was considered missing the maturity that allows it to be integrated within a productive framework. This paper presents an interactive 3D reconstruction workflow for AR applications where each single step is at one of the highest Technology Readiness Level (TRL). The presented workflow is already integrated in a product line and it is essentially based on mature state-of-the-art approaches to perform the capturing of the environment and the generation of its model. To reach the acceptance of the 3D reconstruction procedure by the industrial users, all kind of feedback are provided along this workflow allowing its easy monitoring. In this paper, the different steps of the workflow will be detailed and, all over the sections, practical cues and guidelines to get reconstruction results satisfying the user will be provided.

## 1 INTRODUCTION

It is well known that in vision-based SLAM [4], tracking and reconstruction have a high inter-dependency: to be able to correctly track we need a good 3D reconstruction of the environment and to be able to correctly reconstruct the environment, we need a good tracking. This is still true in the case of AR applications; except that, on top of this, the overlay of the virtual information on the real environment becomes an additional player dependent on both tracking and reconstruction. In fact, to get a consistent and coherent overlay, the tracking as well as the reconstruction of the geometry and the illumination of the environment need to be correctly performed. Once this is achieved, many AR applications could be made possible.

Besides the general AR problem of incorrect occlusions of real objects due to missing depth information (see left image in Figure 1), one major topic of AR applications in industry are measurement tasks between real and virtual objects. There are e.g. different kinds of applications for discrepancy checks, like monitoring the intermediate steps of building a new hall, comparing the current construction plans to the current real state before further conversions are done (see right image in Figure 1) or surveying the correct construction of production facilities [9, 18, 5]. In these applications, the reconstruction of the real environment can serve two purposes: One is providing the data to compare with such that discrepancies can be discovered. The other one is the reverse engineering character, which provides updated data of the current state of the environment, for documentation or for updating of the current construction plans. Another frequently requested task is collision detection and



Figure 1: Applications of reconstructed models (from left to right): Occlusion problem in AR scenarios due to missing depth information of real environment (figures courtesy of Adam Opel GmbH); Collision detection and minimal distance calculation; Discrepancy check (figures courtesy of Volkswagen AG)

minimal distance measurements between the real environment and future objects passing through this environment represented by virtual objects, e.g. changes on products passing the assembly line (see middle image in Figure 1) or in logistics and cargo environment loading of big goods into cargo holds. This inspection provides identification of problematic areas and allows early decisions or replanning.

In the last decades, there has been a bunch of impressive work in geometric 3D reconstruction from a set of images. A quick glance to the evaluation results on the Middlebury Multi-view Stereo Dataset shows the high quality of the 3D reconstruction that one can get using algorithms developed since the publication of the benchmark project paper [14]. In fact, even on challenging dataset, one could get “Super Models” that are very close to the ground truth data. Why the AR industrial user does not use one of these approaches? Because despite their difficulties (lack of texture, auto-occlusion, shadows, ...), first, the datasets do not reflect what a typical industrial AR user (e.g. in an automotive industry, an aerospace industry or working for the army) is aiming to reconstruct. Second, it is often hard to get the desired light conditions and view-angles to hope reaching the precision guaranteed by the evaluated algorithms. Finally, often the background clutter does not allow a simple segmentation of the object to be reconstructed.

We experienced that the same typical industrial AR user when having the choice between such a complete automatic system able to reconstruct many but not all objects of his interest and a system which allows the control over the whole reconstruction procedure and the interaction with the system, he would choose the second option even if it takes much longer. Why? To make sure that it does what he expects.

## 2 RELATED WORK

As many other Computer Vision research topics, image-based reconstruction is becoming incontrovertible for AR. We clearly see it from the recent ISMAR publications that took this direction such as [7, 3, 15, 12]. In [7], the reconstruction is sparse (only feature points are reconstructed) and it is mainly meant to serve the tracking like in traditional vision-based SLAM. In [3], the reconstruction is a goal by itself. A pattern is used for the tracking and a special device is used as a mouse plus a camera in order to allow the user to model the object to be reconstructed. While in [15] the reconstruction is

\*e-mail: marion.langer@metaio.com

†e-mail:selim.benhimane@metaio.com

done in an interactive process mimicking the Match Photo functionality of Google SketchUp and the tracking is markerless. However, in [12], a textured model is created automatically out of the sparse reconstruction done via the SLAM algorithm used for the tracking. Therefore, in this case, the object to be reconstructed needs to be well textured. In the Computer Graphics community, we can mention the work of [17], where a more complex object reconstruction has been achieved and where a larger number of functionalities and forms have been implemented. However, here the reconstruction is done using offline videos. In the Computer Vision community, an impressive work [10] has been recently achieved in the direction of real-time reconstruction. On top of [7], it was made possible to use clusters composed of a set of registered images to get a dense reconstruction of the environment while tracking.

Our approach is definitely not as impressive as the above mentioned papers; it needs higher interaction with the user and it is far simpler. In fact, as we mentioned it in the abstract and in the introduction, our goal is not to compete from the innovation side with these recent approaches. It is essentially to define a workflow based on mature state-of-the-art approaches to perform the generation of the model of the environment. To reach the acceptance of the 3D reconstruction procedure by the industrial users, our workflow had to be clear, completely under the monitoring of the user and should be able to reconstruct any object of his interest with no consideration of the time needed for that. Tracking these objects is not needed and there should be no constraint on their shape, material or color. Advanced methods, like the ones cited above, would possibly be, in later releases, smoothly integrated in the productive workflow as alternatives to speedup the process.

### 3 PREPARATION OF THE INTERACTIVE RECONSTRUCTION

The interactive 3D reconstruction described in this paper is based on a set of registered images of the environment. It requires a calibration of the camera intrinsic parameters and a registration of each image, meaning an estimation of the transformation of the world coordinate system in relation to the camera coordinate system.

#### 3.1 Calibrating the camera

The camera that will be used to acquire images needs to be calibrated for the chosen resolution. The camera calibration can be done e.g. with a 2D calibration pattern of known geometry. Different images of the pattern need to be taken from different viewing angles (we typically use 6 to 12 images) such that every area of the camera image sensor (chip) is covered by the pattern in at least one image. The corners of the pattern in the different images are detected and an estimation of the camera intrinsics is done based on these detected 2D corners and their known planar 3D points [2, 11]. Depending on the accuracy need for the industrial AR application a higher accuracy camera calibration can be performed using a certified 3D calibration grid with accurately measured 3D points as provided e.g. by AICON [1]. The variance of each parameter of the cameras intrinsics can be computed and given to the user as quality measurement for the camera calibration. Both camera calibration approaches showed their reliabilities and are quite standard by now. Once the calibration finished, its settings (resolution, zoom and focus) must be fixed (no unmounting and remounting of the lens).

#### 3.2 Registration and Tracking

To register the captured images to one fixed world coordinate system any tracking system can be used. Since the accuracy of the 3D reconstruction depends on the quality of the registration, a high quality tracking system is needed. If the captured images are used to perform a vision-based tracking, different approaches can be used (like e.g. marker-based, 2D or 3D feature-based both with known geometries, ...). To ensure a high tracking quality while using vision-based tracking systems, the marker-based tracking is

recommended due to the high contrast of the markers, their simple, stable and reliable detection [13]. Markers are printed directly on rigid planes or printed on non glossy paper and then stucked onto rigid planes or planar adapters, which are then attached to the environment. It is important to ensure that the printout fulfills the assumption of the geometry of the marker made in the algorithm (e.g. squared and of some configured size) due to some possible scaling and stretching arising during the printing process. The quality of marker tracking depends in part on the area size of the marker in the captured image. For our marker tracking, a marker area of approximately 40.000 pixels is recommended, which corresponds to an edge length of the marker of about 200 pixels in the image [13].

Multiple markers can be combined representing the same coordinate system, if the environment to be reconstructed (and thus registered) is e.g. very large such that the marker can not be tracked well from any desired perspective or is containing objects which occlude the visibility of just one marker. In the case where all markers are supposed to reference to the same coordinate system, an inter-marker calibration is performed. In the inter-marker calibration, one marker is the reference marker to which all other markers will be registered. For this, the markers in all images of a given image set are tracked and their relative transformations to each other and finally to the reference marker are estimated and optimized over all images. A confidence value corresponding to the residual of the non-linear minimization (we use Levenberg-Marquardt) is given to the user. If it is too high, additional images may be needed. The image set should contain images with at least two detectable markers each. For each marker to be registered to the reference marker there must be a chain of images which provides a transformation chain from the marker over possibly other markers to the reference marker. The result of the inter-marker calibration is a set of transformations between markers and is not related to the camera used. Therefore, it can be done with a different camera resolution or a completely different camera than the one used for the interactive reconstruction. It is highly recommended to perform the inter-marker calibration on high resolution images captured with a well-calibrated camera.

#### 3.3 Acquiring images for interactive 3D reconstruction

After the tracking system is set up in the real environment, pictures need to be taken with the calibrated camera. The images should be taken from different views. Every vertex to be reconstructed should be visible in at least two images of different views. To get a good reconstruction, many views from wide baselines are needed. However, to allow an automatic matching of a user specified 3D vertex in other images of the scenario (see section 4.1.2), it is additionally recommended to take a set of pictures in which translations parallel to the surface of the vertices are performed. For texturing the model, there should be one fronto-parallel image to every reconstructed face.

### 4 OVERVIEW OF THE INTERACTIVE 3D RECONSTRUCTION

The interactive 3D reconstruction is embedded in an AR-System [8], which is capable of loading and displaying images, performing undistortion of each image based on a given camera calibration, performing registration of each image based on a given tracking configuration, and overlay (render) virtual objects on the images according to the tracking results.

The model generation is based on the creation of faces of the environment to reconstruct. To generate a face, its vertices first need to be reconstructed. Based on these vertices, the faces are specified and can be textured afterwards. The interactive 3D reconstruction offers an interface, in which the user can monitor and interact with the current state of reconstruction:

The reconstructed vertices (see section 4.1) are presented in a tree structure: "3D points overview" (see left image in Figure 2).

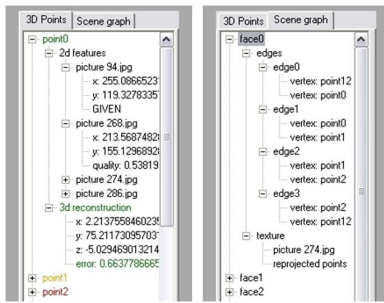


Figure 2: Interface of interactive reconstruction. Tree structure of 3D point reconstruction (left) and face reconstruction (right)

Every vertex consists of an ID, a set of 2D features and a 3D reconstruction. The 2D features consist of their image and their (u,v) pixel position in the image. For each 2D feature that was automatically matched (see 4.1.2) its matching quality is stated. The 3D reconstruction consists of the (x,y,z) values once a vertex is reconstructed. Additionally, it will be augmented in the renderer. For the 3D reconstruction of a vertex, its root-mean-square (RMS) reprojection error is stated as well as quality feedback of the reconstruction.

The reconstructed surfaces (see section 4.2) are presented in another tree structure: "faces overview" (see right image in Figure 2). Every face consists of an ID, an ordered set of edges and a texture. Each edge is made up of two reconstructed vertices. The texture is pointing to an image used for texturing. Each edge and reconstructed face will be augmented in the renderer. For displaying reconstructed faces, backface-culling is used. Thus, only the front sides of the faces are rendered.

#### 4.1 3D reconstruction of vertices

To reconstruct a vertex, it needs to be specified in two images. To specify a point in an image there are two possibilities.

##### 4.1.1 User defined points by clicking in the image

The point needs to be selected from the 3D points overview. The image containing the point needs to be loaded. By clicking onto the point displayed in the render window, the point is specified. The image name and the (u,v) coordinates of the clicked point will be added to the selected points "2D features" in the 3D points overview (see Figure 2). The 2D feature will be marked as "GIVEN" to indicate that it was a reliable user defined point.

##### 4.1.2 Automatic matching

After a point was specified in at least one image the point can be automatically matched in the other images of the same folder. The automatic matching is performed for all user specified 2D features of the currently loaded image based on a mature approach (see Figure 3): To match a 2D feature from one image to another, the patch around the 2D feature of specified size is searched in the other image. The sum-of-square-differences (SSD) or the normalized cross-correlation (NCC) can be used as similarity. Since no big illumination changes are expected between two images of the same image set, our system uses the SSD comparison. To reduce the search of the patch, it is only searched along the epipolar line of the features point in the other image. To simplify the search along the epipolar line to a 1D-search, the images are first rectified [6]. The two patches with the smallest SSD-values are set into relation. If the one with the smallest SSD is significantly smaller than the second smallest, the former one will be considered as match. To provide quality feedback of the matching, the NCC of the two patches will

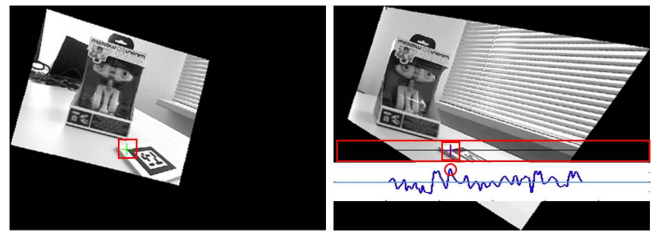


Figure 3: Automatic matching: The 2D feature of one image (left) is searched along the 1D-epipolar line in the rectified other image (right). The SSD and NCC values are computed and the best values corresponding pixel is kept as match candidate. (The graph in the right image shows the NCC values along the epipolar line)

be computed. After the automatic matching is performed, the results will be added to the point in the 3D points' overview and the quality of each matching is stated (see Figure 2).

##### 4.1.3 Automatic 3D reconstruction of specified points

As soon as two 2D points are specified in two images, the 3D points is reconstructed with a simple but reliable approach: First the 3D viewing rays are computed for each image in which a 2D feature of the point is known. The 3D viewing ray is the vector from the camera center of the registered image through the 2D feature in the image. The 3D vertex corresponding to the optimal ray for all viewing rays results in the 3D reconstruction of the vertex [16]. As feedback of the quality of reconstruction the RMS-error is computed between the reprojected reconstructed 3D vertex into the single images and their known 2D positions.

The resulting reconstruction will be rendered (see Figure 4) and its (x,y,z)-coordinates will be displayed together with the RMS reprojected error in the 3D points overview (see Figure 2). For intuitive feedback of the quality, the vertices are colored in accordance to the RMS reprojection error in red (high error ( $> 2$  pixel)), orange (medium error ( $> 1$  pixel)) or only two 2D features used for reconstruction) or green (small error ( $< 1$  pixel)). These empirical values should be related to the actual image resolution.

#### 4.2 3D reconstruction of surfaces

To reconstruct an object, the 3D reconstructed vertices are used to create the faces of the object. A valid face is defined by the following rules: (1) It consists of at least three different vertices. (2) The vertices are given in the order of their appearance in the faces border. The faces normal is defined by the counter-clockwise order of its vertices. (3) The starting and the ending vertex of the face are the same. (4) It lies in a plane: If more than three vertices specify the face, they do not need to lie exactly in a plane, but if it is not the case the reconstruction might differ from the users' intention.

To reconstruct a face, 3D reconstructed vertices are selected in counter clock wise order and added one after the other to the face. Two consecutive vertices specify an edge of the face which are created automatically (see right image in Figure 2). Selecting the 3D vertices can be performed in the 3D points overview or in the render window. The rendering can also be set to a pure Virtual Reality mode (VR mode) with free camera navigation, which allows a more convenient selection of points. The reconstructed faces are directly displayed in the render window (see Figure 4).

#### 4.3 Texturing of surfaces

A reconstructed face can be textured by an image selected by the user. As feedback to the user, it is added to the faces overview (see Figure 2). The vertices of the reconstructed face are reprojected into the image to define the polygonal region of interest in the image used for texturing of the face. This image region is copied

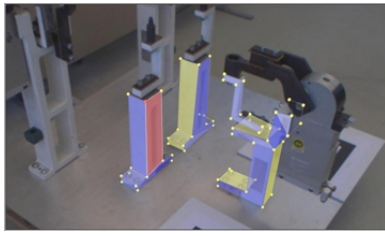


Figure 4: Reconstructed 3D vertices and faces are visualized

into a texture map and the new coordinates of the vertices of the polygon are computed and stored as texture coordinates for each vertex of the face. Alternatively, the image to texture a face could be automatically selected. For this, all reprojected vertices of the face to texture must lie in the image. To ensure that the face is not back-facing the camera, the orientation of the face to the camera is computed based on the scalar product between the normal unit vector of the face and the unit vector parallel to the optical axis of the camera. The image in which the area of the reprojected face is the biggest and front-facing the camera is chosen for texturing.

#### 4.4 Export of the results

Finally, the reconstruction is exported to be used for further tasks (see Figure 5). The textured 3D model can be exported in VRML 2.0 format. This will allow the model to be stored and imported into other tools for tasks like reverse engineering, more realistic AR scenarios due to correct occlusions or measurements (like collision detection, minimal distance, ...) between virtual and reconstructed real environment. Additionally the reconstructed 3D vertices can be exported in an own text file storing the x, y and z coordinates of the 3D vertices and its given name. This data can be used for 3D/3D registration and further 3D point based algorithms e.g. measurements, like point based discrepancy check. Finally the working state of reconstruction can be exported in an xml file and imported for later continuation of reconstruction.

#### 5 CONCLUSION

In this paper, we presented a very basic but reliable and complete workflow to generate 3D models of basically any environment. The approach is highly interactive and allows the user to control and monitor each step of the workflow and the reconstruction. Feedback about the quality of the different steps are provided to the user. Mature approaches were integrated for the different steps in the workflow. In this first approach, we did not focus on the needed time for performing a reconstruction, but on controlling and monitoring the reconstruction process. The resulting workflow and software will be used as a basis on which “smarter” algorithms will be plugged in.

#### REFERENCES

[1] AICON 3D Systems GmbH. <http://www.aicon.de/>.  
 [2] J. Y. Bouguet. Camera calibration toolbox for matlab. <http://www.vision.caltech.edu/bouguetj/calib.doc/>.  
 [3] P. Bunnun and W. Mayol-Cuevas. Outliner: an assisted interactive model building system with reduced computational effort. In *ISMAR*, 2008.  
 [4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *PAMI*, 26(6):1052–1067, 2007.  
 [5] P. Georgel, P. Schroeder, S. Benhimane, S. Hinterstoisser, M. Appel, and N. Navab. An Industrial Augmented Reality Solution For Discrepancy Check. In *ISMAR*, 2007.  
 [6] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2003.

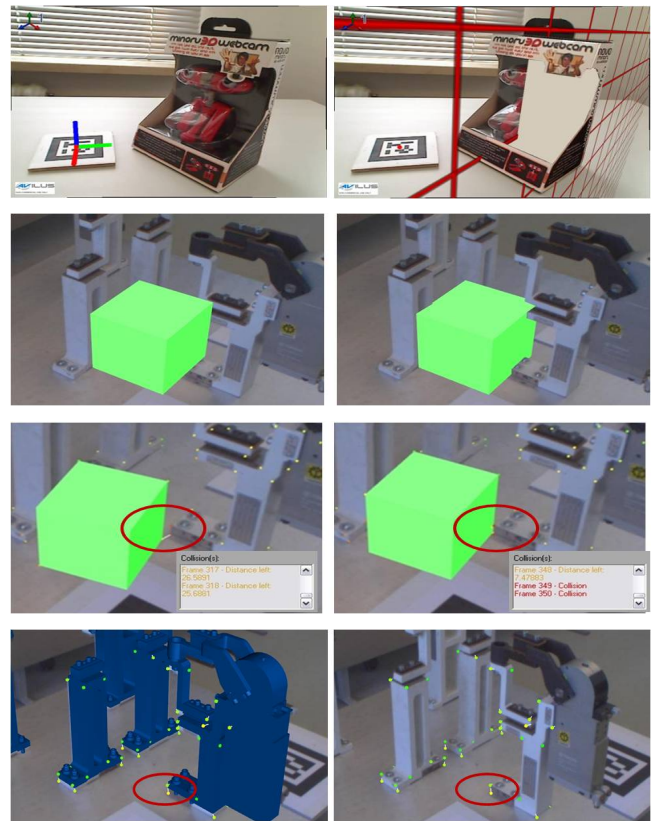


Figure 5: Results (from top to bottom): Reverse engineered box is compared against the real; Correct occlusion due to reconstruction; Minimal distance and collision detection; Discrepancy check

[7] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *ISMAR*, 2007.  
 [8] metaio GmbH. [www.metaio.com](http://www.metaio.com).  
 [9] N. Navab, N. Craft, S. Bauer, and A. Bani-Hashemi. Cylicon: software package for 3d reconstruction of industrial pipelines. In *IEEE Workshop on Applications of Computer Vision*, 1998.  
 [10] R. A. Newcombe and A. J. Davison. Live dense reconstruction with a single moving camera. In *CVPR 2010*, pages 1498 – 1505, 2010.  
 [11] OpenCV. <http://opencv.willowgarage.com/wiki/>.  
 [12] Q. Pan, G. Reitmayr, and T. W. Drummond. Interactive model reconstruction with user guidance. In *ISMAR*, 2009.  
 [13] K. Pentenrieder. *Augmented Reality based Factory Planning*. PhD thesis, Technical University of Munich, 2008.  
 [14] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR 2006*, 2006.  
 [15] G. Simon. Immersive image-based modeling of polyhedral scenes. In *ISMAR*, 2009.  
 [16] G. Slabaugh, R. Schafer, and M. Levingston. Optimal ray intersection for computing 3d points from n-view correspondences. Technical report, 2001.  
 [17] A. van den Hengel, A. Dick, T. Thormählen, B. Ward, and P. H. S. Torr. Videotrace: rapid interactive scene modelling from video. In *SIGGRAPH 2007*. ACM, 2007.  
 [18] S. Weibel, M. Becker, D. Stricker, and H. Wuest. Identifying differences between cad and physical mock-ups using ar. In *ISMAR*, 2007.