Modeling, Tracking, Annotating and Augmenting a 3D Object in less than 5 Minutes

Steve Bourgeois∗, Boris Meden†, Vincent Gay-Bellile∗, Mohamed Tamaazousti∗ and Sebastian Knodel†

∗CEA LIST, Vision and Content Engineering Laboratory, Point Courrier 94, Gif-sur-Yvette, F-91191 France,
Email: name.familyName@cea.fr
†Diotasoft, Massy, France
Email: name.familyName@diotasoft.fr

Abstract—This article presents a comprehensive augmented reality solution designed for professional uses. Recently, we introduced a new high-end visual tracking solution based on the concept of SLAM constrained with a 3D mesh model. Here, we push the concept even forward, proposing a complete framework that addresses the whole pipeline, from object modeling to its tracking and providing the augmentations.

The solution being designed to target professional applications, it provides high robustness and accuracy in the tracking as well as an easy and fast deployment process. Moreover, this solution is completed by a new AR device, embedding an HD camera with buttons in a gun-style wearable device with a display deported on a flat screen. This device was developed to answer the ergonomic requirements in particular use cases such as quality control or workforce education. A video presenting the framework is available at [1].

I. INTRODUCTION

During the last years, augmented reality applications have known increasing developments in the publishing and advertising industry. This is mostly due to the introduction of Augmented Reality frameworks [2], [3], [4], [5] that meet both user and industry requirements. Indeed, these frameworks rely on robust and accurate planar surface tracking algorithms that provide a great user experience. Moreover, it also includes intuitive tools for content creation and application deployment, reducing both production costs and time to market.

To cover more domains and continue its expansion, Augmented Reality lacks offrameworks that handle 3D objects with a quality and a productivity equivalent to those dedicated to planar objects.

Indeed, the development of such a framework is facing several challenges. The most obvious is the need of a robust and accurate 3D object tracking. However, as the tracking process relies on a modelisation of the object of interest, the ability to produce such a model by a non expert in reduced time must also be considered. The last challenge is relative to the content creation of the AR application. If contents creation tools for planar objects [6] were inspired by Desktop Publishing software (DTP) to facilitate their integration in the publishing industry process, a similar reflection is needed for 3D objects context in order to provide a more intuitive and efficient interface than those of usual 3D authoring tool.

In this paper, we introduce an Augmented Reality framework for 3D object that tackle these different challenges. After a short state of the art (II) , an overview of our framework is exposed (III). Then a brief presentation of the tracking module is given (IV), followed by the presentation of a toolset that allows to both deploy the tracking (V) and create the AR content (VI) in few minutes. Finally, a series of experiments is conducted to validate the whole pipeline (VII).

II. PREVIOUS WORK

Currently, most of the AR frameworks for 3D objects [3], [7] rely on a model based tracking solutions. Among this approach, two families of model can be distinguished [8]: purely geometric model and photo-geometric model (also called appearance model). The former relies exclusively on a geometric representation of the object (usually, a 3D mesh) while the latter combines both geometric and appearance information (usually, a point cloud with local appearance descriptors).

Currently, nearly all commercial AR frameworks are based on photo-geometric model[3], [7]. For a long time, this choice was justified by the better robustness and deployment simplicity of the photo-geometric solutions. Indeed, 3D mesh model were expensive to build, and geometric tracking processes were usually highly sensitive to model imperfections and complex camera motions. Contrariwise, since the construction of a photo-geometric model is based on multi-view geometry, it requires only an inexpensive camera. Even if the first solutions were time consuming, requiring to instrumentalize the environment [9] or the intervention of an expert [10], many improvements have been provided to simplify and accelerate this step. The main
one consists in using monocular visual SLAM [11] since it allows the user move freely around the object with a standard camera without any preparation.

However, deployment cost of the tracking solution should not be the only criterion for the choice of an AR Framework: content creation cost should also be considered. Whereas 3D point cloud models are inexpensive to produce, referencing 3D content in such models is far more complex than using a 3D mesh since perception of a point cloud is unusual for a human operator. To simplify this task for 3D point cloud models, two approaches were introduced. The first one consists in aligning the 3D point cloud with the CAD model of the object. This is usually manual task achieved by the user through dedicated interface. Some semi-automatic solutions were introduced to reduce the number of user interaction and improve the point cloud accuracy [7]. Nevertheless, such approaches requires a CAD model whereas photo-geometric model were favored because they did not require such model. To prevent this drawback, the second solution consists in using interactive modeling tools to reconstruct a coarse 3D mesh from a video sequence. This modelization can be done off-line [12] or on-line [13], [14]. Since the resulting 3D mesh is not used for the tracking, accuracy is no longer an issue. Nevertheless, most of these solutions require expert user.

While these different tools reduce the deployment cost, the photo-geometric approach remains subject to major drawbacks. First, textureless objects cannot be represented by such model since resulting point cloud would be too sparse. Second, such photometric solutions are extremely sensitive to variations of the lighting conditions. When the lighting conditions are discordant with the object’s model built off-line, there is no way to bootstrap the tracking process.

Recently, we introduced a new tracking solution that combines purely geometric model with visual SLAM [15] to provide both an accurate and robust tracking for a large variety of objects. In the following, we demonstrate that our solution take benefit of geometric tracking solutions properties without their disadvantages in term of deployment.

III. SYSTEM OVERVIEW

Our framework is constituted of 3 modules, each one responding to a specific challenge:

- **Tracking and Mapping Module**: includes accurate and robust 3D object tracking algorithms that provide in real-time the motion of the camera with respect to the object of interest;
- **Deployment Module**: provides both a toolset for fast and easy 3D object modeling, and different tracking initialization solutions;
- **User Interface & Content Management Module**: provides a set of software and hardware interfaces that allow the user to create and consult the virtual content of the scene.

These modules and their interconnections, summarized in figure 2, are described in the following sections. A video presentation of the framework is available at [1].

IV. ROBUST AND ACCURATE 3D LOCALIZATION

Our tracking solution relies on two process:

- the tracking process, which estimates the motion of the camera from an initial pose;
- the relocalisation process, which provides the initial pose to bootstrap the tracking process after a failure.

These two process, described in the following sections, rely on the hypothesis that the object is static wrt. its environment.

A. Tracking with Constrained SLAM

Visual Simultaneous Localization And Mapping [16], [17] is a process that provides, in real-time, an estimation of the relative motion of a camera and a 3D map of the surrounding environment. Since the reference coordinate frame and scale factor of the map are arbitrary chosen, this process is not adapted to object tracking. Nevertheless, we demonstrate in our previous work [15] that SLAM can be constrained by a purely geometric model to provide a robust and accurate object tracking solution.

Given an initial pose of the camera (cf. section IV-B and V-B), the tracking process is initialized by projecting the 2D feature points of the image on the 3D mesh model, providing a initial 3D map that share the coordinate frame and scale factor of the 3D mesh model. The following image are localized with a traditional keyframe-SLAM approach except at the keyframe. When a keyframe is created, the sharp edge of the model are projected with the pose estimated by the SLAM process and matched with the nearest contour of similar orientation[15]. During the bundle adjustment, both reprojection errors of the 3D map points and 3D edges are minimized to improve both the 3D points map and the camera trajectory[15].

Our constrained SLAM solution provides several benefits with respect to usual geometric object tracking. First, it is able to provide an accurate localization even if the object is occluded or out of the field of view. Indeed, the 3D point map of the environment is not only optimal with respect to multiview geometry but also with respect to the constraints provided by the 3D mesh model. Therefore, the
3D point cloud and the 3D mesh share the same coordinate frame and scale factor. Consequently, as long as the object is static with respect to the scene, this 3D point cloud can be used to localize the object even if this one is occluded or out of the field of view. Second, our solution is robust to large motion, imperfect 3D mesh model and coarse initialization. Indeed, while usual geometric tracking process require small motion to achieve the edge-to-contour matching, our solution is robust to large and complex motions since the edge-to-contour matching use the pose predicted by the SLAM process. Moreover, using the environment of the object to estimate the camera trajectory allows to avoid local minima introduced by model and initialization inaccuracies. Thus a highly accurate 3D mesh model and an exact initialization are no longer necessary.

B. On-line appearance modeling for fast reinitialization

To bootstrap the tracking process after a failure, a viewpoint recognition algorithm [18] is used to provide the initial pose. The indexed viewpoint are provided by 3D map and keyframes of the previous tracking sessions. Indeed, the tracking process provide a 3D map similar to the appearance models used in other AR framework (cf. section II). And since this model was built few seconds ago, it is rational to consider the lighting condition and surrounding environment unchanged. Therefore, this model is more relevant than an appearance model built offline a long time ago and limited to the object interest.

V. DEPLOYMENT TOOLS

This section introduces the tools and process that allows the deployment of the tracking solution. These tools are designed to require the minimum of time and user expertise. Since the camera calibration process is quite standard, we will focus on the creation of the tracking model (V-A) and the initialization of the tracking algorithm (IV).

A. Easy and fast 3D object modeling

Our tracking algorithm requires a 3D model to estimate the camera trajectory. If an accurate CAD model is preferable, it is frequently unavailable: they might not exist (eg. craft or artistic object) or can be unreachable (eg. restricted access due to confidentiality). In these cases, our framework includes an application that allows to reconstruct a 3D mesh of the object on the fly. As illustrated in figure 3, this application is a 3 step process:

- Initial reconstruction: provides an initial reconstruction of the object and its surrounding environment;
- Object segmentation: extracts the 3D object mesh from the initial reconstruction and defines its coordinate frame;
- Model simplification: realizes a simplification of the object mesh to provide a tracking model compatible with real-time performances.

The first step relies on the KinectFusion process [19]. This solution has been selected for its deployment speed and the quality of the reconstruction it provides without requiring neither user expertise nor expensive hardware. Moreover, this solution is able to model a large variety of objects, including textureless objects, and can be extended to cover large volume [20].

To be exploitable by the tracking process, the surrounding environment must be removed from the model, remaining only the geometry of the object of interest. To achieve this task, the second step of the modeling process assumes that the object of interest lies on a flat surface. Because the scene may include several planar surfaces, the user identifies this particular plan by putting a 2D marker (eg. a coded target) on its surface. Assuming this marker was observed by the 3D camera during the initial reconstruction step, its location and orientation in the reconstructed model coordinate frame can be estimate [21]. The reconstruction can then be expressed in the coordinate frame of the marker, providing a non-arbitrary coordinate frame that will be used in section V-B. Since the equation of the planar surface is known, points located under this plane are suppressed and the remaining points are partitioned into entities by a clustering algorithm. The object model is then identified as the most frequently observed cluster during the first reconstruction step.

The ultimate step consists in simplifying the object model to reach a number of polygon compatible with real-time performances. This is achieved using a Quadric Edge Collapse decimation algorithm.

B. Initialization

Besides the 3D object modeling, a coarse first camera localization is required to bootstrap the tracking. To provide

\(1\)The plane equation \(z = 0\) can be refined by selecting the surrounding point cloud and using a plane fitting algorithm.
this initialization, various solutions are possible depending if
the object has already been tracked in this environment or not.

In the first case, an appearance model of the object was
produced during the previous tracking session (cf. §IV-B). Therefore, in the same way as other AR frameworks, this
appearance model can be used by the viewpoint recognition
algorithm introduced in section IV-B to initialize the tracking
process.

In the second case, because the object’s appearance model
is unavailable or do not match the current lighting conditions,
viewpoint recognition is irrelevant. However, since the tracking
is robust to inaccurate initialization, any solution providing
a coarse initialization can be used to bootstrap the tracking
process. For example, since the object’s model is expressed
in the 2D maker coordinate frame, it is sufficient to place this
marker approximately at the same position with respect to the
object. Various strategies relying on user interaction can also
be used to provide this coarse initialization.

Whatever initialization used, the following tracking session
provides a new map of the scene that can be used to create or
update the appearance model of the object (cf. §IV-B). Unlike other AR framework, it becomes therefore conceivable
to deploy the system in different places or with different
lighting conditions.

VI. USER INTERFACES FOR CONTENT MANAGEMENT

For an AR framework, user interfaces are critical for both
content production and consultation. A first challenge is to
design a content creation tool that minimizes the time required
to conceive AR scenarios. A second challenge is to provide an
user-friendly interface to improve the adoption of the system
by the end-user.

A. In Situ Interactive Content Creation

In industry context, the 3D content of an AR application
should be both fast and develop and ergonomic for the end-user.
While most of the AR framework rely on classic 3D authoring
tools, such as Unity3D, Blender, etc., we consider these tools
do not perfectly fulfill these requirements. First, such tool does
not provide a real WYSIWYG experience since the designer’s
viewpoint do not match with the natural viewpoint of a end-
user. It is therefore difficult to assess the ergonomics of the
resulting content. Second, these tools do not provide a natural
and intuitive interface for 3D object manipulation and 3D
navigation, making the production process time consuming and
expansive even for simple content.

To tackle these issues, our framework is provided with
an intuitive interface for 3D object annotation in Unity3D.
It relies on an in situ interactive content creation process.
Similar to other in situ content creation tools [22], annotations
are created online while the object is tracked (cf. §IV-V). The
3D location of the annotation is determined through a point-
and-shoot interface metaphor: the user indicates in the online
video stream a 2D point of the object, the corresponding 3D
location on the object’s surface is determined by raycasting,
and the annotation’s location is finally defined from this 3D
point and an offset along the normal of the surface. Once the
3D location is defined, the user can choose the object to pin
among a 3D assets. This extensible 3D object library includes
a visual vocabulary (cf. figure 4) for standard actions (push,
pull, turn, screw,...) and objects (paper, video clap, picture,...).
Moreover, each 3D annotation can be linked to 2D content
such as video, text, images, website,...

By using an in situ approach, our solution provides a
natural 3D navigation interface since the operator moves in the
real space instead of a virtual space. Moreover, the point-and
shoot interaction allows to define the location of the annotation
in a single interaction with an intuitive interface. And finally,
since our solution provides an AR visualization of the content
similar to the one experimented by the end-user, the operator
can easily assess the ergonomics of its work. This approach
make the content creation process easier and faster, reducing
the cost of development. Moreover, it allows the industrial to
capitalize its existing 2D documentation and therefore reduces
the impact on the workflow. If such approach is limited to
simple content, the remaining complex content can be achieved
with the usual interface of the 3D authoring tool.

B. Ergonomic user interface

Currently, almost all AR framework are restricted to be
used on a tablets or smartphones because of the liberty of
motion it provides. Nevertheless, such hardware provides a
poor grip, a limited autonomy, a low image quality (rolling
shutter) and field of view, and a low robustness to shock.
Moreover, such solution is not relevant for all use case, such
as collaborative AR experience that require the users to share
the same display.
For all of these reasons, on top of the tablet introduced in [23],
we introduce a new device named Selltic Wand (cf. figure 4).
Compared to the former device, the latter is mostly designed

Fig. 4. 3D content creation while tracking. First, the user point a location and
choose the content to add through a contextual menu (a). The corresponding
3D entity is then added at the indicated location and displayed in AR(b).

Fig. 5. Selltic AR devices family. Left: Selltic Wand and its Asus
XTion™ scanner plugin. Right: Selltic Tablet

2What You See Is What You Get
for collaborative AR experience, such as workforce education, selling assistance in a showroom, or exhibition. It includes an HD camera to provide high quality image (global shutter, wide field of view) compatible with a large screen display. It also integrates a joystick for menu navigation, and a trigger and multiple buttons for validation. The shape was designed to provide a good grip and receive a pluggable XTion scanner for the reconstruction process (cf. §V-A).

During an AR experience, the optical axis of the camera is used as a target. Therefore, the device can be used as a gun, aiming for an annotation with the optical axis and shooting with the trigger to access to the linked content. To improve the accuracy, the joystick can be used to achieve a numeric zoom on the displayed image.

VII. Experiments

To evaluate our framework, 2 series of experiments are conducted. The first one compare the registration quality using a CAD model and the model reconstructed with our framework. The second one aims to evaluate the time required to deploy a new AR scenario on an unknown object.

A. CAD model vs. reconstructed model

Obviously, the 3D model provided by the reconstruction process (cf. section V-A) is subject to artifacts. Indeed, even if the reconstruction is relatively accurate (cf. figure 6), sharp edges are rounded while reconstruction noise may induce some fictive sharp edges. The objective of this experiment is therefore to evaluate the impact of these artifacts on the localization process.

Two mechanical objects with their associated CAD model were selected for this experiment (see figure 8): a car cylinder head and a part of an exo-skeleton. These objects were reconstructed with the algorithm introduced in section V-A using an Asus XTion 3D camera. Since we do not dispose of a trajectory ground truth, the effects of reconstruction artifacts on the tracking process is assessed by comparing the localization of the camera reached with the CAD model and the reconstructed model.

Figure 7(a) represents the obtained trajectories on the cylinder head sequence with the two types of models while figure 7(b) represents the distribution of the 3D deviation between the two the trajectories. The mean deviation obtained on the whole sequence of the cylinder head is of 6 mm (see figure 7) for a 30 cm object observed from about 1m away. Note that similar results have been obtained on the exo-skeleton but are not presented here as not to overload the paper. Nevertheless, figure 8 illustrates the registration quality for the two object and for the both kind of model.

Therefore, the impact of the reconstruction artifact on the registration can be considered as negligible.

B. Timing evaluation

Production and deployment time are crucial features for an AR framework. Therefore, we evaluate on a small set of users (10 users) the time required to design a new AR application from scratch. This experiments is achieved on a generator and is considered as achieved when 3 annotations are added. The timing of each step of the process are measured at the 5th attempt.

From these experiments, it appears that the whole process require approximately 5 minutes. The 3D object modeling requires approximately 3 minutes whereas 2 minutes are necessary to produce the annotations. The timing of the initialization is null since it relies on the target used during modeling.

VIII. Conclusion

In this article, we exposed our analysis concerning the design of an AR framework for 3D object and introduced our solution. To be easily integrated in an industry workflow, our framework includes the tools required from the production to the deployment of an AR application. Indeed, while our solution provides a robust and accurate tracking process that exploits a 3D mesh model, it also provides a tool to reconstruct such a model in few minutes, different initialization strategies, and ergonomic hardware/software user interfaces.

Compared to other 3D object tracking solutions, our approach is not only more accurate and robust, but also easier and faster to deploy. Indeed, if most of 3D object tracking frameworks are based on an appearance model of the object of interest that requires to be rebuilt when the lighting conditions change, our tracking process relies on a purely geometric model that is insensitive to lighting conditions. Therefore, our object model is built only once without requiring expensive hardware or user expertise. Moreover, because the tracking process is robust to coarse initialization, our framework is provided with a set of initialization solutions, allowing to adapt the initialization strategy to the application context. The resulting framework can be use in various context, such as workforce education (cf. video [1]) or selling aid tools (cf. video [24]).

Future work will target to realize a larger ergonomic evaluation of the framework. Reconstruction of larger objects or areas will also be studied.

REFERENCES

Fig. 6. Comparison of CAD model (a & c) and reconstructed models (b & d) for a cylinder head and a part of an exo-skeleton. The hue of reconstructed model represents the surface deviation to the CAD model, while gray areas represent elements that were missing in the CAD model.

Fig. 8. Registration with a CAD model (first row) and with the reconstructed model (second row). The 3D mesh is represented in blue.