Deformed Reality

Antoine Petit, Nazim Haouchine, Frédérick Roy, Dan B Goldman, Stéphane Cotin

EG UK Computer Graphics & Visual Computing (2019)
G. K. L. Tam and J. C. Roberts (Editors)

Deformed Reality

Antoine Petit[1], Nazim Haouchine[1], Frederick Roy[1], Dan B. Goldman[2] and Stephane Cotin[1†]

[1]Inria, [2]Google Inc

(a) (b) (c)

Figure 1: *Deformed Reality: Our framework allows a user to intuitively interact with an augmented reality scene by producing rigid and deformable 3D transformations while preserving visual consistency.*

Abstract

We present Deformed Reality, a new way of interacting with an augmented reality environment by manipulating 3D objects in an intuitive and physically-consistent manner. Using the core principle of augmented reality to estimate rigid pose over time, our method makes it possible for the user to deform the targeted object while it is being rendered with its natural texture, giving the sense of a interactive scene editing. Our framework follows a computationally efficient pipeline that uses a proxy CAD model for both pose computation, physically-based manipulations and scene appearance estimation. The final composition is built upon a continuous image completion and re-texturing process to preserve visual consistency. The presented results show that our method can open new ways of using augmented reality by not only augmenting the environment but also interacting with objects intuitively.

1. Introduction

The considerable technical advances in augmented reality (AR) made it possible for users to have different sorts of interactions such as object grasping, rigid transform or surface deformation [CH13]. However, most of the time, a 3D mesh is superimposed using computer-generated texturing, making the final output unrealistic. In the context of video and image editing, recent methods using 3D stock models to replace the actual object projection in the image have been proposed [KSES14] [ZCC*12] [CZS*13] [HRC*18]. These methods give the possibility of editing objects in a scene in 3D and extending the range of manipulation. However, they are essentially dedicated to photo editing or assume no relative motion between the object and the camera when dealing with videos. This makes it difficult to be used for AR purpose where object pose has to be estimated over frames.

In this work, we aim at enabling user-generated manipulations of moving objects in their environment, without producing visual breaks due to 3D overlays. To this end, we introduce the concept of Deformed Reality, a method for deforming objects in a video stream during an AR process. This method relies on a rigid frame-by-frame pose estimation technique to overcome related works' fixed camera assumption, combined to a real-time physics-based simulation that enables consistent non-rigid manipulations. The final composition is rendered using the original object texture making

† stephane.cotin@inria.fr

the whole output realistic. Moreover, we makes it easy for users to realize complex edits during AR with little manual intervention and interactive feedback, and is based on two main contributions:

- A simultaneous simulation-and-tracking pipeline capable of transferring a finite element simulation onto a video while preserving the spatial and visual coherence of the original video.
- Allowing for a wide range of edits, including rigid transforms, elastic deformations, and internal parameter updates such as mass, stiffness or gravity.

2. Related Works

Manipulating objects during AR involves very similar algorithms as those used in image and videos editing. This domain has gained interest from the research community with the prominence of commercial tools and the extended use of smart-phones and hand-held cameras. One of the most active area of research is editing human faces which has received a wide attention with the Face2Face method [TZN*15; TZS*16]. A tool for photorealistic edits of facial expressions in videos in real time, based on the low-dimensional parametric face model has been proposed by Blanz and Vetter [BV99]. Human shapes have been addressed for images [ZFL*10] and videos [JTST10], enabling quick and easy manipulation of human bodies in 2D content based on an underlying 3D morphable model. Bai *et al.* [BAAR12] focus on temporal edits by selectively de-animating videos. The idea is to compute a warp field constrained by a set of manually-provided strokes that removes the large-scale motion in these regions while leaving finer-scale, relative motions intact. Haouchine *et al.* [HDK*12] advocate the use of physics-based models to capture deformable objects in a video stream during AR. This method uses stereoscopic features tracking to compute a 3D displacement field that is applied back to the superimposed 3D model. Bazin et al. [BYM*16] also use physics simulation to edit videos as a way to prevent object parts with no texture information from becoming visible. The 3D models are essentially modeled around the input video, limiting the complexity of their results. Physics-based modeling is also used by Paulus et al. [PHCC15] to permit topological changes during augmentation of deformable objects. Their method uses visual tracking to detect cutting and fracture to pilot a physical model while being overlaid onto the video. Interactive Dynamic Video was proposed by Davis et al. [DCD15]. Although it produces videos, it takes as input a single image where image-space representation of an object is extracted from vibrations. These vibrations are use to synthesize physically plausible animations and permits manipulation on unseen physical forces without knowledge of the scene geometry or material properties. In the context of 3D image manipulation, Zheng et al. [ZCC*12] use cuboid proxies to edit a scene in a variety of editing tasks; e.g., replacing all furniture in a living room with different models and embed them (naturally) in the image. Following this direction Chen et al. [CZS*13] propose 3-sweep, an interactive editing tool to extract 3D objects from a scene in a quick, intuitive manner. It facilitates the extrusion of coarse 3D shapes from images with only three sweep strokes, each defining a dimension. Although limited to rigid transformations only, it represents a powerful editing tool to extracted objects and associated parts. Kholgade et al. [KSES14] utilize 3D CAD models proxies, obtained from Internet database as a prior. They estimate illumination and texture after aligning the 3D model in addition to re-texturing hidden areas using a symmetric-based algorithm. A new photograph is composed by interacting with the 3D model transferred onto the image. Haouchine et al. [HRC*18] extended this work with more manipulations, including deformation, cutting and non-rigid interaction, in addition to simplifying the 3D/2D initialization with a user-centered contour-based fitting algorithm. This method is however limited to fixed camera which makes it not practical for AR applications.

Besides object manipulation, video editing can be used to update viewpoint [CPW*11] [OCDD01] or edit timelines [LZW*13]. In [CPW*11] the authors propose a method based on depth layers which enables viewpoint changes in videos where, as in [LZW*13], objects can be moved in time while keeping their original position, giving the user the ability to manipulate the video timeline.

Most of these previous works focused on photographs and rigid object editing, while the few ones dealing with actual videos assume static scenes where no relative motion between the object and the camera is considered. Deformed Reality overcomes these assumptions and permits manipulating objects in a video stream, in the presence of camera (or object) motion. This interactive editing can be performed in parallel of camera pose estimation and 3D model transfer onto the augmented view.

3. Overview

Our computational pipeline is separated into three simultaneous processes, illustrated in Figure 2: (1) given a video and a 3D model, we estimate the object pose by tracking in a rigid manner the object over frames, (2) we compute the deformation based on the user input using an underlying physical model and (3) we back-project the 3D edited object onto an inpainted frame following a continuous completion and re-texturing process. The 3D CAD proxy model, used as input, is provided by the user and can be retrieved from various databases.

4. Model-based Rigid Pose Estimation

The issue of estimating the complete rigid 3D pose of the camera, with respect to the considered object along the image stream (cf Figure 3), is addressed based on the real-time 3D model-based tracking approach presented in [PMK13][PMK14]. The method consists in combining, within a deterministic optimization process, geometrical information provided by tracked edge features, with a dense color information through object / background color separation, to efficiently handle 3D models of any shape and complexity.

4.1. A local non-linear minimization problem

Given a 3D model of the object, the goal is to estimate, frame-by-frame, the rigid camera pose $\mathbf{r}$, which transforms the object frame into the camera frame, by minimizing the function $\Delta(\mathbf{r})$ accounting for errors $e_i(\mathbf{r})$ between a set of visual features extracted from the image and the forward projection of their 3D homologous in the image plane:

$$\Delta(\mathbf{r}) = \sum_i \rho(e_i(\mathbf{r})) \quad (1)$$

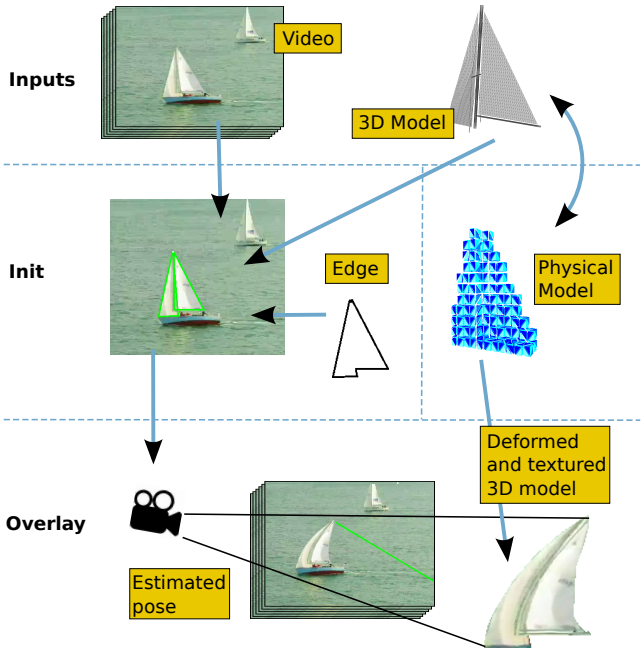


Figure 2: *Deformed Reality pipeline: using as inputs a video and a 3D proxy CAD corresponding to an object in the scene, the user starts by aligning the 3D model onto the image. The initial pose is then estimated along with the object's edges. These edges will be used to compute a rigid object pose over frames. Simultaneously, a deformable model is built from the 3D CAD surface mesh to enable non-rigid object manipulations. The deformed mesh is textured and projected onto an inpainted image using the estimated pose to produce the final composition.*

where ρ is a robust estimator, which reduces the sensitivity to outliers. This is a non-linear minimization problem with respect to the pose parameters $\mathbf{r}$, and we follow a Gauss-Newton minimization framework to handle it. At each iteration k in the process, a displacement is performed in the parameter space which is $SE(3)$:

$$\mathbf{r}_{k+1} = \mathbf{r}_k \oplus \delta\mathbf{r} \tag{2}$$

where $\oplus$ is an internal compositional law in the parameter space ($SE(3)$). The displacement $\delta\mathbf{P}$ is computed through:

$$\delta\mathbf{r} = -\lambda(\mathbf{DJ})^{+}\mathbf{De} \tag{3}$$

$$\text{with} \quad \mathbf{J} = \frac{\partial\mathbf{e}(\mathbf{r})}{\partial\mathbf{r}} \tag{4}$$

and where $(\mathbf{DJ})^{+}$ is the pseudo inverse of $\mathbf{DJ}$, $\mathbf{J}$ being, as written above, the Jacobian matrix of the error vector $\mathbf{e}(\mathbf{r})$ with respect to the pose. λ is a proportional gain and $\mathbf{D}$ is a weighting matrix associated to the Tukey robust estimator. Based on equation (2), the pose $\mathbf{r}$, represented by its homogeneous matrix ${}^{c}\mathbf{M}_o$, can be updated as follows, using the exponential map [MSKS04]:

$${}^{c}\mathbf{M}_o^{k+1} = exp([\delta\mathbf{r}])\ {}^{c}\mathbf{M}_o^{k}. \tag{5}$$

where $exp([\delta\mathbf{r}])$ is the exponential map enabling to express the rigid motion $\delta\mathbf{M}$ generated by $\delta\mathbf{r}$ at iteration k.

4.2. Visual cues

In order to benefit from the complementarity of different visual features and to overcome the limitations of classical single cue approaches, we integrate in the computation of Δ a geometrical edge-based function Δ^g and a color-based one Δ^c with the corresponding weights w^g, and w^c:

$$\Delta = w^g\Delta^g + w^c\Delta^c \tag{6}$$

Δ^g and Δ^c are computed in a similar way to [PMK13][PMK14]. Δ^g relies on line-to-point correspondences between model and image edges and on geometrical distances accounting for these correspondences. Edges have the advantage of being robust to illumination conditions but suffer from having similar appearance, resulting in ambiguities and potential local minima.

For Δ^c, the goal is to avoid any image extraction or segmentation that could lead to outliers and mismatch. By modeling the color (or luminance) appearance on both sides of the silhouette edges of the projected 3D model using simple statistics, a better accuracy can be achieved, with the advantage of being robust to image or motion blur, background clutter or noise.

4.3. Efficient processing of the complete CAD model

One of the drawbacks of classical 3D model-based methods [CMPC06; DC02; PMK11] is that all the segments making of the 3D polygonal model are treated. It implies dealing with simple objects or performing manual and heavy pre-processing on the CAD model to make it compliant with real-time or computationally efficient implementations. The approach used in this work considers the direct use of a complete, but not necessarily polygonal model, which can be textured or untextured.

For efficiency, we employ a 3D rendering engine to efficiently handle 3D projections, determining visible salient edges from the rendered scene, and use these in the processing of the visual cues described in section 4.2. This enables our system to handle potentially very large models.

An advantage of this technique is to automatically handle the hidden face removal process and to implicitly handle self-occlusions. The whole set of edges and their 3D homologues, retrieved through the rendered depth buffer, will be used to compute Δ^g, while Δ^c only considers the silhouette edges.

4.4. Filtering

We incorporate a Kalman filtering process to smooth pose estimates and reduce jittering. As in [PMK14], we employ a linear Kalman filter on the camera velocity parameters $\mathbf{v}$, which is then integrated to determine the pose so that: ${}^{c}\mathbf{M}_o^{i+1} = exp([\mathbf{v}^{i+1}\delta t])\ {}^{c}\mathbf{M}_o^{i}$, at each time step i. We assume a constant velocity model. The camera pose and the camera displacement between successive frames can be assumed to be random variables, following Gaussian distributions. Through their covariances matrices, their uncertainty can be characterized, propagated from the low-level uncertainty, giving an indicator on the reliability of the tracking process and providing a global observation noise covariance matrix $\Sigma_{\mathbf{P}}$, which feeds the computation of the Kalman gain (see [PMK14] for more details).

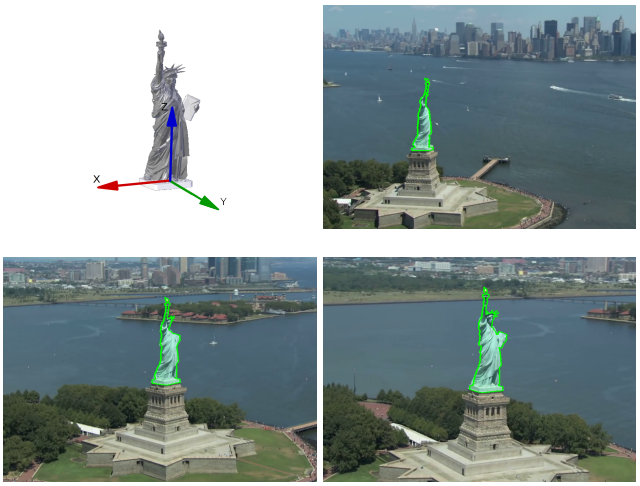


Figure 3: *Pose Estimation: selected frames from the video stream showing the mesh overlaid on the input image (green contour) from camera pose estimation.*

5. Deformable Manipulation

We rely on physics-based modeling to enable deformable manipulation [MDM*02]. In our method, physical models are used to interactively manipulate objects in a wide range of 3D distortions including stretching, torsion, compression. In addition, the user can interactively edit physical attributes like stiffness, mass or gravity.

Using the input 3D model M, we first build using voxelization a volume $\mathbf{V}$ composed of co-rotated elements [NMK*06]. These elements represent the Degrees-of-Freedom (DoF) of the physical model following a finite element representation (cf Figure 4). The number of elements chosen in a tradeoff between interactive performance and sufficient accuracy. This process is done once at initialization.

A particular object deformation is specified by the displacements of nodal positions and the nodal forces. We build a stiffness matrix $\mathbf{K}$ depending on object's material properties, Young's modulus and Poisson's ratio. This enables us to linearize the relationship between nodal forces and nodal positions and permits real-time computation.

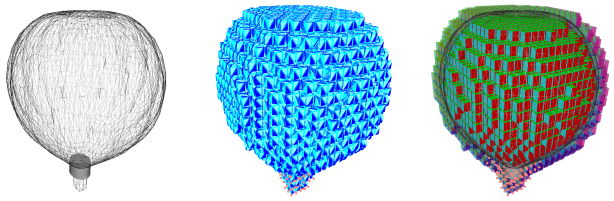
Finally, fixed boundary conditions $\mathbf{q}$ are necessary to correctly model the deformations. These constraints represent a part of the 3D model that is considered fixed.

We denote $\Theta(\cdot)$ as the function that transforms an input 3D model M into a physical model D so that:

$$\mathsf{D} = \Theta(\mathsf{M}, \mathbf{V}, \mathbf{K}, \mathbf{q}) \tag{7}$$

Once the volume built following $\Theta(\cdot)$, the stiffness defined and the constrained regions chosen, the user can interact with the object at each frame of the sequence by moving the DoF producing forces that will deform the object according to its stiffness.

In addition to rigid transforms our framework enables object manipulation in a 3D manner including stretching, torsion, and compression. The produced deformations highly depend on the values of E the Young's modulus, and μ the Poisson ratio. For users not accustomed to using physical engines, pre-defined parameters are set according to object size and units. We also consider topological changes such as cutting or fracturing, and collisions that are detected by computing a proximity distance between two or more objects in the scene. Several type of forces can be applied to simulate pressure, wind or friction and act as external force on mesh surface. Physical attributes, whether intrinsic (related to the object, like stiffness, mass, and damping), or extrinsic (related to the scene, like gravity, or external forces), can be updated directly during the augmentation. The user can thus simulate deflation and melting as well as scene dynamics like falling under high gravity or large mass.



(a) (b) (c)

Figure 4: *Finite element volume and boundary constraints: the volumetric mesh* $\mathbf{V}$ *is built upon the surface model M in (a) following a grid tetrahedron volume model (b), or a grid hexahedron volume model (c). The red spheres represent the boundary constraints.*

6. Composition

In order to produce a final composition that is similar to the input sequence our framework measures scene lighting parameters and object material following a minimization scheme while object's background is replaced using inpainting after removing the original element from the scene. Both steps are described above.

6.1. Scene Appearance

Visually realistic augmentation requires knowledge of the object's diffuse properties and scene's light parameters. Inspired by [HNl05], we use a simplified Torrence-Sparrow reflection model that defines the specular reflection of an object's surface point as

$$\mathsf{I}^c(i) = \left[\frac{k_{d,c}, \cos\theta_i}{r^2} + \frac{k_{s,c}}{r^2 \cos\theta_r} \exp\left[\frac{-\alpha^2}{2\sigma^2}\right]\right] \text{with} c \in \{r,g,b\} \tag{8}$$

where $\mathsf{I}(i)$ is the i^{th} image pixel value, θ_i the angle between the light source direction and the surface normal, θ_r is the angle between the viewing direction and the surface normal, α is the angle between the surface normal and the intersection of the viewing direction and the light source direction. r represents the distance between the light source and the object surface point, k_d and k_s are coefficients for the diffuse and specular reflection components respectively and include light source intensity, and σ is the surface roughness.

We want to estimate an ensemble of parameters that consists of

the specular reflection properties (k_s,σ), diffuse reflection k_d and light source position r from image I and the registered 3D mesh M. To do so, we start by directly calculating θ_r, α and θ_i using our inputs. First, the angle θ_r can be obtained using the registered geometry M and camera position obtained from 3D/2D registration, then assuming a unique light source and a convex object, light source direction can be estimated by back-projecting image specular peaks on geometry normals which permits to estimate α and θ_i.

Assuming a Lambertian material with constant albedo, we follow a diffuse-based constraints scheme to first estimate r knowing k_d then we refine for (k_s,σ) to finally solve for (r,k_d,k_s,σ) minimizing the squared error as

$$\underset{r,k_d,k_s,\sigma}{\operatorname{argmin}} \sum_{i\in\chi} \tau_i \Big(\mathsf{I}(i) - \Big[\frac{k_d \cos\theta_i}{r^2} + \frac{k_s}{r^2 \cos\theta_r} \exp\Big[\frac{-\alpha^2}{2\sigma^2}\Big]\Big]\Big)^2 \quad (9)$$

where I_i the image pixel value of i and τ_i is a compensation factor used to avoid image compensation when computing the residual error. The domain χ represents the region of interest for the optimization scheme, where the diffuse image is used to estimate light position and diffuse reflection where the original image will be used for specular reflection estimation.

6.2. Image Completion and Re-texturing

Producing the final composition requires continuously building a new image I_c as a result of the projection of the deformed and textured 3D model on the input image I.

First an inpainted image I_p is generated using image completion technique following the silhouette contour of the projection of the 3D model with respect to the pose $\mathbf{r}$.

We used the PatchMatch iterative algorithm [BSFG09]. In order to speed-up the completion process, we only compute one iteration of the algorithm. Moreover, since most of the time the manipulations do not imply all the object, the area of the revealed part to compensate can be reduced to the difference between the original and the projected element, making the computation faster.

Once the completion is computed, the output composite image is built using the projection function Ω following the expression:

$$\mathsf{I}_\mathsf{c}{}^f = \Omega\big(\mathbf{P}^f, \mathscr{M}(\mathsf{D}^f, \mathbf{T}^f), \mathsf{I}_\mathsf{p}{}^f\big) \quad (10)$$

where $\mathbf{T}$ is the texture of the targeted object, $\mathscr{M}(\cdot)$ is a texture mapping function, f is the frame, I_p is the inpainted image and $\mathbf{P}^f$ represents the 3D/2D camera projection matrix from the object frame to the image plane.

7. Results

We tested our method on four examples, a Yacht sailing, an aerial view of the Statue of Liberty, a farmer with a Hat ploughing the ground and an Air balloon gliding through the air. Figure 8 shows the obtained results.

The 3D models were obtained from Internet databases (freely available). The physical models are automatically generated using a sparse hexahedral or tetrahedral grid fitting the object surface, independently of the surface discretization. The number of nodes

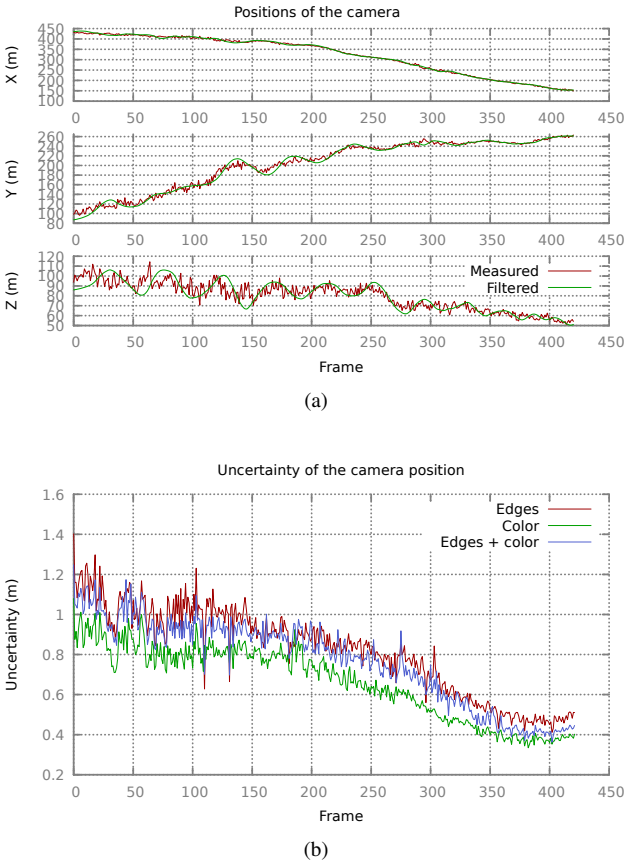


Figure 5: *Camera pose over frames for the statue example: (a) The measured and filtered camera translation in the object frame (from ${}^o\mathbf{M}_c$) in X, Y and Z. (b) The uncertainty indicator over the camera position estimation, computed as the trace of the covariance matrix of the estimated pose ${}^c\mathbf{M}_o$ (for the position parameters), for the two visual cues (edge and color) independently and jointly.*

(nodes) of the finite element model, the stiffness E, Poisson ratio ν, and number of constrained nodes (fixed nodes) used for each scenario are reported in Table 1. These parameters can be tuned as desired by the user, and pre-defined parameters are automatically set according to the 3D model size. We used the framework Sofa [†] for physical simulation. Zero displacement constraints are manually set on the physical model and represent the "boat pivot" for the Yacht, the "head" for the Farmer's Hat example, the "basket" for the Air balloon example and the "pedestal" for the Statue of Liberty.

In all examples, the object or the camera is moving. After a manual initial alignment, the pose is estimated over frames. Since a part of the object may be fixed (in object coordinates), the estimated pose is loaded on the virtual camera (in world coordinates). We used OpenGL for rendering.

Results for the tracking and pose estimation process are featured

[†] www.sofa-framework.org

Table 1: User-selected physical parameters for each example.

	E (Kpa)	ν	Nodes	Fixed Nodes
Yacht	450	0.45	192	22
Statue of Liberty	100	0.45	6750	225
Airballoon	25	0.45	288	16
Farmer's Hat	5000	0.3	1000	120

in Figure 5, with visual overlays showing the accuracy the method. The plots of the positions of the camera in the object frame demonstrates the benefit of filtering the complete pose to provide smooth estimates, which is essential to get visually realistic re-textured re-projections, as described in Section 6.2. Note that the camera parameters are unknown for the presented examples and are manually set, explaining the the potentially physically non-realistic estimations of the pose, and demonstrating the robustness of the pose tracking system w.r.t calibration of the camera.

We show several user interactions: stretching and pulling the Yacht and the Hat, inflating and deflating the AirBalloon, and more complex elastic deformations of the Statue of Liberty. Depending on the number of nodes of the 3D model, the process can run at interactive rates, reaching up to 17 fps.

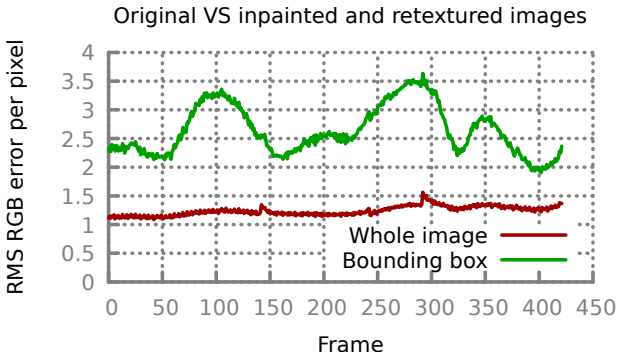


Figure 6: *Evaluation of the image in-painting and re-texturing method by comparing the original image with the re-texturing one (statue example), with the mesh rigidly reprojected given the estimated pose. It is evaluated on the whole image (red) and on a bounding box around the projected model (green).*

As an evaluation of the image completion and re-texturing method we show in Figure 6 the error between the original image and the in-painted and re-textured one, with the rigidly reprojected mesh, given the estimated rigid pose. The metric is the root mean square RGB error per pixel, evaluated on the whole image, and also over a bounding box around the silhouette of the projected mesh, here for the scenario of the Statue of Liberty. It shows the sensitivity of the image completion and re-texturing method especially with cluttered background, for instance when the statue passes over the island behind (frames 250-330).

Figure 7: *Our approach may fail to correctly estimate the pose. Detection of contours may fall into ambiguities since edge/background separation is based on gradient saliency.*

8. Limitations

When dealing with videos with dynamic backgrounds, a temporal and coherent in-painting method is necessary to produce a final composition without visual break. The completion process considered in this work does not include temporal consistency and is computed frame-by-frame. However, we believe that using physics-based models and scene dynamics can bring useful information to the establishment of a temporal inpainting, where object position can be predicted in time, and their silhouette can be used to compensate the new revealed parts of the video.

Another limitation of our framework resides in the robust estimation of a 3D pose, since the pose estimation may fail depending of object's background and in the presence of large occlusions (see Figure 7). This can be damaging to the whole pipeline since it also creates the region of completion and the detour the re-texturing area. We can think of recent methods based on trained histograms [TSS17] that could be plugged to our framework. Ideally, we would like to have a 3D abstraction of the scene making the alignment fully automatic and more accurate. Approaches proposed in [WL15], [SQLG15], [KMT*17] would then be useful solutions to provide pose (re-)initialization to our pipeline. Our work could also benefit from recent works based on machine learning [CXG*16], [XGF16], where 3D reconstructions are estimated using learned models that extract shape priors from large collection of 3D data, or more recent work using convolutional neural networks [ISS17]. Another improvement would be to generate the 3D model directly from the input stream, using recent structure-from-motion methods [OKI15], or by relying on intuitive user-centered methods such as *3-sweep* [CZS*13] or *cuboids* [ZCC*12] for simple geometry.

9. Conclusion

We presented "Deformed Reality" a new paradigm for augmented reality that enables the user to deform objects in scenes without producing visual breaks usually emanating from 3D mesh overlay in classical AR systems. Using an efficient method to estimate object pose and relying on a physics-based model built on-the-fly, the final composition is rendered sequentially to produce a sequence where objects are deformed and registered in real-time giving sens of manipulating the surrounding of the objects. Several limitations are to be addressed to bring our concept to a well-established technique mainly: (i) a robust real-time continuous inpainting, (ii) handling occluded regions during pose estimation and (iii) appearance and

Figure 8: *3D physics-based manipulations of various objects in using Deformed Reality pipeline. The user can interact with objects through a rich range of manipulations, including rigid and non-rigid transforms and physical parameter updates such as mass, stiffness or gravity, and thereby pull on the the Yacht's sail while moving in a race, deform the Farmer's hat with elastic response, deform the air balloon and make it deflate, inflate and fall under gravity and melt the Statue of Liberty. [video sequences are shown in the supplemental video]*

illumination estimation from images under arbitrary conditions. With this in mind, we strongly believe that "Deformed Reality" can open new ways in making AR more interactive and enhance user experience with its surrounding.

References

[BAAR12] BAI, JIAMIN, AGARWALA, ASEEM, AGRAWALA, MANEESH, and RAMAMOORTHI, RAVI. "Selectively de-animating video." *ACM Trans. Graph.* 31.4 (2012), 66–1 2.

[BSFG09] BARNES, CONNELLY, SHECHTMAN, ELI, FINKELSTEIN, ADAM, and GOLDMAN, DAN. "PatchMatch: a randomized correspondence algorithm for structural image editing". *ACM Transactions on Graphics-TOG* 28.3 (2009), 24 5.

[BV99] BLANZ, VOLKER and VETTER, THOMAS. "A morphable model for the synthesis of 3D faces". *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co. 1999, 187–194 2.

[BYM*16] BAZIN, J-C, YU, GUO, MARTIN, TOBIAS, et al. "Physically Based Video Editing". *Computer Graphics Forum*. Vol. 35. 7. Wiley Online Library. 2016, 421–429 2.

[CH13] CHUN, WENDY H. and HÖLLERER, TOBIAS. "Real-time Hand Interaction for Augmented Reality on Mobile Phones". *Intelligent User Interfaces*. Santa Monica, California, USA: ACM, 2013, 307–314. ISBN: 978-1-4503-1965-2 1.

[CMPC06] COMPORT, A.I., MARCHAND, E., PRESSIGOUT, M., and CHAUMETTE, F. "Real-time markerless tracking for augmented reality: the virtual visual servoing framework". *IEEE Trans. on Visualization and Computer Graphics* 12.4 (July 2006), 615–628 3.

[CPW*11] CHEN, JIAWEN, PARIS, SYLVAIN, WANG, JUE, et al. "The video mesh: A data structure for image-based three-dimensional video editing". *Computational Photography (ICCP), 2011 IEEE International Conference on*. IEEE. 2011, 1–8 2.

[CXG*16] CHOY, CHRISTOPHER B., XU, DANFEI, GWAK, JUNYOUNG, et al. "3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction". *ECCV 2016*. 2016, 628–644. ISBN: 978-3-319-46484-8 6.

[CZS*13] CHEN, TAO, ZHU, ZHE, SHAMIR, ARIEL, et al. "3-Sweep: extracting editable objects from a single photo". *ACM Transactions on Graphics (TOG)* 32.6 (2013), 195 1, 2, 6.

[DC02] DRUMMOND, T. and CIPOLLA, R. "Real-Time Visual Tracking of Complex Structures". *IEEE Trans. on Pattern Analysis and Machine Intelligence* 24.7 (July 2002), 932–946 3.

[DCD15] DAVIS, ABE, CHEN, JUSTIN G, and DURAND, FRÉDO. "Image-space modal bases for plausible manipulation of objects in video". *ACM Transactions on Graphics (TOG)* 34.6 (2015), 239 2.

[HDK*12] HAOUCHINE, NAZIM, DEQUIDT, JFFFDFFFDRFFFDFFFDMIE, KERRIEN, ERWAN, et al. "Physics-based Augmented Reality for 3D Deformable Object". *Workshop on Virtual Reality Interaction and Physical Simulation*. The Eurographics Association, 2012. ISBN: 978-3-905673-96-8 2.

[HNl05] HARA, K., NISHINO, K., and LKEUCHI, K. "Light source position and reflectance estimation from a single view without the distant illumination assumption". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.4 (Apr. 2005), 493–505. ISSN: 0162-8828 4.

[HRC*18] HAOUCHINE, NAZIM, ROY, FREDERICK, COURTECUISSE, HADRIEN, et al. "Calipso: physics-based image and video editing through CAD model proxies". *The Visual Computer* (Oct. 2018). ISSN: 1432-2315 1, 2.

[ISS17] IZADINIA, HAMID, SHAN, QI, and SEITZ, STEVEN M. "IM2CAD". *CVPR*. 2017 6.

[JTST10] JAIN, ARJUN, THORMÄHLEN, THORSTEN, SEIDEL, HANS-PETER, and THEOBALT, CHRISTIAN. "Moviereshape: Tracking and reshaping of humans in videos". *ACM Transactions on Graphics (TOG)*. Vol. 29. 6. 2010, 148 2.

[KMT*17] KEHL, WADIM, MANHARDT, FABIAN, TOMBARI, FEDERICO, et al. "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, 1521–1529 6.

[KSES14] KHOLGADE, NATASHA, SIMON, TOMAS, EFROS, ALEXEI, and SHEIKH, YASER. "3D Object Manipulation in a Single Photograph Using Stock 3D Models". *ACM Trans. Graph.* 33.4 (July 2014), 127:1–127:12. ISSN: 0730-0301 1, 2.

[LZW*13] LU, SHAO-PING, ZHANG, SONG-HAI, WEI, JIN, et al. "Timeline Editing of Objects in Video". *IEEE Transactions on Visualization and Computer Graphics* 19.7 (July 2013), 1218–1227. ISSN: 1077-2626 2.

[MDM*02] MÜLLER, MATTHIAS, DORSEY, JULIE, MCMILLAN, LEONARD, et al. "Stable Real-time Deformations". *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '02. San Antonio, Texas: ACM, 2002, 49–54. ISBN: 1-58113-573-4 4.

[MSKS04] MA, Y., SOATTO, S., KOŠECKÁ, J., and SASTRY, S. *An invitation to 3-D vision*. Springer, 2004 3.

[NMK*06] NEALEN, ANDREW, MÜLLER, MATTHIAS, KEISER, RICHARD, et al. "Physically based deformable models in computer graphics". *Computer Graphics Forum*. Vol. 25 (4). Wiley Online Library. 2006, 809–836 4.

[OCDD01] OH, BYONG MOK, CHEN, MAX, DORSEY, JULIE, and DURAND, FRÉDO. "Image-based modeling and photo editing". *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM. 2001, 433–442 2.

[OKI15] ONDRUSKA, PETER, KOHLI, PUSHMEET, and IZADI, SHAHRAM. "MobileFusion: Real-Time Volumetric Surface Reconstruction and Dense Tracking on Mobile Phones." *IEEE Trans. Vis. Comput. Graph.* 21.11 (2015), 1251–1258 6.

[PHCC15] PAULUS, C. J., HAOUCHINE, N., CAZIER, D., and COTIN, S. "Augmented Reality during Cutting and Tearing of Deformable Objects". *2015 IEEE International Symposium on Mixed and Augmented Reality*. Sept. 2015, 54–59 2.

[PMK11] PETIT, A., MARCHAND, E., and KANANI, K. "Vision-based Space Autonomous Rendezvous : A Case Study". San Francisco, USA, Sept. 2011, 619–624 3.

[PMK13] PETIT, A., MARCHAND, E., and KANANI, K. "A robust model-based tracker for space applications: combining edge and color information". *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'2013*. Tokyo, Japan, Nov. 2013 2, 3.

[PMK14] PETIT, ANTOINE, MARCHAND, ERIC, and KANANI, KEYVAN. "Combining complementary edge, keypoint and color features in model-based tracking for highly dynamic scenes". *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, 4115–4120 2, 3.

[SQLG15] SU, HAO, QI, CHARLES R, LI, YANGYAN, and GUIBAS, LEONIDAS J. "Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views". *Proceedings of the IEEE International Conference on Computer Vision*. 2015, 2686–2694 6.

[TSS17] TJADEN, H., SCHWANECKE, U., and SCHFFFDFFFDMER, E. "Real-Time Monocular Pose Estimation of 3D Objects Using Temporally Consistent Local Color Histograms". *2017 IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017, 124–132 6.

[TZN*15] THIES, JUSTUS, ZOLLHÖFER, MICHAEL, NIESSNER, MATTHIAS, et al. "Real-time expression transfer for facial reenactment". *ACM Transactions on Graphics (TOG)* 34.6 (2015), 183 2.

[TZS*16] THIES, JUSTUS, ZOLLHÖFER, MICHAEL, STAMMINGER, MARC, et al. "Face2face: Real-time face capture and reenactment of rgb videos". *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* 1 (2016) 2.

[WL15] WOHLHART, PAUL and LEPETIT, VINCENT. "Learning descriptors for object recognition and 3d pose estimation". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 3109–3118 6.

[XGF16] XIE, JUNYUAN, GIRSHICK, ROSS, and FARHADI, ALI. "Deep3D: Fully Automatic 2D-to-3D Video Conversion with Deep Convolutional Neural Networks". *ECCV 2016*. Cham, 2016, 842–857. ISBN: 978-3-319-46493-0 6.

[ZCC*12] ZHENG, YOUYI, CHEN, XIANG, CHENG, MING-MING, et al. "Interactive images: cuboid proxies for smart image manipulation." *ACM Trans. Graph.* 31.4 (2012), 99–1 1, 2, 6.

[ZFL*10] ZHOU, SHIZHE, FU, HONGBO, LIU, LIGANG, et al. "Parametric Reshaping of Human Bodies in Images". *ACM Trans. Graph.* 29.4 (July 2010), 126:1–126:10. ISSN: 0730-0301 2.