RGB-D based tracking of complex objects

Alejandro Perez-Yus[1], Luis Puig[2], Gonzalo Lopez-Nicolas[1], Jose J. Guerrero[1], and Dieter Fox[2]

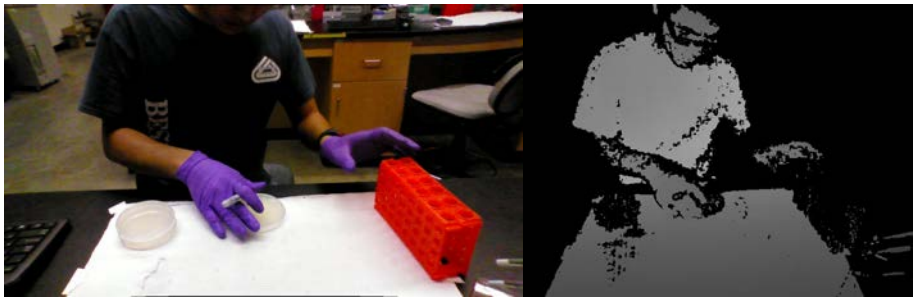
[1] Universidad de Zaragoza, Spain
{alperez, gonlopez, josechu.guerrero}@unizar.es
[2] University of Washington, Seattle, USA
{lpuig, fox}@cs.washington.com

Abstract. Tracking the pose of objects is a relevant topic in computer vision, which potentially allows to recover meaningful information for other applications such as task supervision, robot manipulation or activity recognition. In the last years, RGB-D cameras have been widely adopted for this problem with impressive results. However, there are certain objects whose surface properties or complex shapes prevents the depth sensor from returning good depth measurements, and only color-based methods can be applied. In this work, we show how the depth information of the surroundings of the object can still be useful in the object pose tracking with RGB-D even in this situation. Specifically, we propose using the depth information to handle occlusions in a state of the art region-based object pose tracking algorithm. Experiments with recordings of humans naturally interacting with difficult objects have been performed, showing the advantages of our contribution in several image sequences.

1 Introduction

Object detection and tracking have always been an important topic in robotics and computer vision. Knowing the 6D pose of an object in real-time can be helpful for many applications such as robot manipulation, online task supervision or action recognition. The variability and complexity of the objects and scenes makes this problem hard to solve with computer vision and, therefore, it remains relevant in the field. Lately, thanks to the popularization of consumer depth cameras such as Microsoft Kinect, new algorithms have been proposed, including the tracking of articulated objects [24] with the addition of physical constraints [23]. However, these works rely uniquely on the depth measurements, which may not be suitable for certain situations when the depth information is not reliable. For example, when tracking what we call *complex* or *difficult objects* (e.g. those with thin or intrincate shapes, made of some types reflective surfaces or glass objects) the depth measurements provided by conventional RGB-D cameras cannot be used for this purpose. An example of these type of objects is shown in Fig. 1.

In this work, we focus on tracking robustly the 6D pose of difficult objects. The considered framework is a single RGB-D camera, using the RGB component whenever the depth camera fails to recover meaningful information. We studied previous approaches of 6D object tracking using different cues, specially paying attention to those methods using RGB cameras. We chose as starting point one of these state of the art methods



(a) (b)

Fig. 1: Sample frame extracted from the video sequences used in this work, including color image (a) and depth image (b). In (b) the tones of grey vary with the depth and black pixels are pixels with no depth information. Some objects such as the red tube holder or the circular plates are not visible in the depth image due to the reflective properties of their materials. These are the type of objects we consider in this work.

[21], which uses region-based object tracking with 3D models. Then, we perform enhancements to adapt the method to the particular problem considered here. Specifically, we explore how the depth image around the object can be used to improve the tracking even when the object itself is hardly perceivable. By using depth information, we show that it is feasible to detect the points in the scene that are visually occluding the object in its current estimated pose. Taking advantage of knowing the occluding region, we modify the optimization performed by [21] to improve the object tracking avoiding failures derived from occlusions.

The motivation of this work is the object tracking during human manipulation. In our dataset, the humans are performing a laboratory activity, acting unaware that they are being recorded and producing natural video sequences (a frame of one of these sequences shown in Fig. 1). The working environment is supposed to be known, so we have previous CAD models of the objects in the scene. We have used a dataset of such conditions to perform experiments, showing how our approach outperforms the default method. We prove how this mutual collaboration between color and depth information overcomes limitations that pure depth-based methods and pure color-based methods have by design: color enables complex object tracking for the former and depth helps handling occlusions for the latter.

2 Related work

There are many different approaches to tackle object pose tracking problem with computer vision [16, 27]. Some methods work better with specific type of objects. For example, when the objects are highly textured and the texture is known, a way to retrieve the pose can be to find correspondences between the template and the image (using, for instance, point correspondences), and then using geometric methods such as homogra-

phies to compute the pose. These methods work well in "cereal box" type of objects [10, 14], but are prone to fail in textureless objects, as the features extracted change with perspective or illumination. Given that we focus on difficult objects that usually lack prominent textures, these previous methods are considered out of scope for our problem.

Some other authors consider the problem simultaneously with the segmentation of the object in the RGB image given a known 3D model [22, 4, 8, 21]. These methods are called *region-based* methods. They work with the fact that (usually) objects have a distinct property that differenciates them from the background, such as the color. A function is defined so that it returns the likelihood of every pixel to belong to the background or the foreground (i.e. the object). The pose of the object is optimized so that its projection in the image leaves foreground pixels inside and background pixels outside following an energy minimization problem. These methods have severe limitations. For instance, this approach is prone to fail when there is no clear distinction between the foreground and the background (e.g. transparent objects, similar colors), and has problems handling occlusions. Besides, sometimes the poses that optimize the problem might not be unique. However, it has the advantage of working with rapidly moving objects (which may appear blurred in the image).

Alternative approaches are the so-called *edge-based* methods. They are based in the extraction of edges in the image, which are usually invariant to illumination, and appear in both textured and textureless objects even when they are transparent. Essentially, these methods consist in finding the pose of the object that makes its 3D edges (as projected from the CAD model) match its correspondent edges extracted in the image by using some contour extraction algorithm. Some examples of this kind of approach are [9, 26]. For the tracking, Particle Filter has been used in some works such as [3, 7]. Inspired by [17], Imperioli et al. [13] do not only use the location of the edges themselves but also the orientation in what is called Directional Chamfer Distance (DCD). Edge-based methods often have initialization problems, but for tracking applications with small displacement between frames, they should be robust enough if the edges are well detected (i.e. no motion blur). However, these methods are usually rather slow for real-time applications, and it is often hard to match edges correctly in highly cluttered environments, where wrong matches may occur.

There are some works that specifically focus on the detection of transparent objects. For instance, [15] use the infrared emitter and sensor from a ToF camera to detect and reconstruct transparent objects. A RGB-D camera is used in [19] to estimate the object pose. In [20], they use a stereo camera to look for transparent objects in the observed inverse perspective mapping discrepancy.

We can also add some physical constraints to guide the retrieval of the 6D pose to plausible object positions. For instance, two different volumes cannot intersect, including the user hands and the working table. This can be done reasoning with the volumes themselves via TSDF [23] or by using physics engines such as Bullet [11, 25] or MuJoCo [18]. Gravity can also be included in the system with an appropriate physics engine.

A combination of methods could be the best approach to tackle this problem, since every one of them has its own advantages and limitations. Some existing approaches

that combine several cues are the following: Edge-based and color-based approach is presented in [2]. Keypoints are used in combination with edges [6] or 3D point clouds [1]. One of the most popular approaches in object detection is the LINE-MOD [12]. It combines both depth and edge-based cues in a unified framework. Seo et al. [5] uses region-based knowledge to extract confident searching directions in order to enhance their edge-based approach.

In this work, we propose a combination of methods, using RGB and depth with a single RGB-D device. Unlike other proposals, our main interest is to focus in those objects which reflect depth very badly. Nevertheless, the depth information of the surroundings can still be used to improve the object localization, e.g. segmenting the background information, estimating the work table plane or tracking the hands of the user. We propose to add this information extracted from the depth to enhance a color-based pose tracking algorithm. The idea is to complement existing methods which use depth cameras in those objects whose depth information cannot be retrieved. As the dataset we use have some fast movements, we decided to start from [21], which is able to handle this circumstance successfully. Then, we use the depth information to solve one of its main limitations: the occlusions.

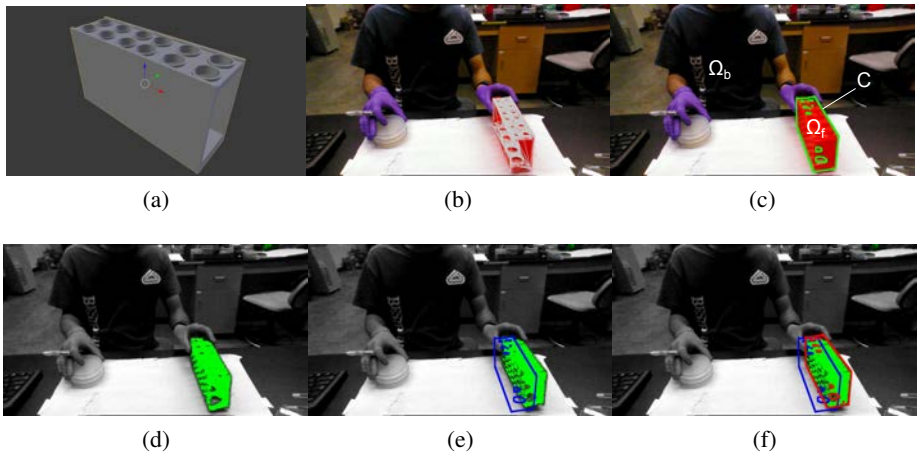
3 Proposed Method

In this work, our goal is to use a RGB-D camera to successfully track objects which cannot be properly perceived by the depth sensor. This means that the observation of the object needs to be performed with the color camera instead. However, our proposal shows that the depth information outside the object can still be helpful in this context. For this, we choose to start with a state of the art method for object tracking with color images, the PWP3D [21]. The original method is briefly described in Section 3.1 Then, in Section 3.2 we explain how we have enhanced the method by including depth information to the algorithm.

3.1 Object tracking with PWP3D

The PWP3D is a method for real-time segmentation and tracking of 3D objects [21]. It assumes a 3D model of the objects to track is available, which is true in restricted environments such as the one considered in this work (Fig. 2a). The problem consists in retrieving the pose of the object that makes the projection of the 3D model match pixel by pixel its image representation along a sequence of images (Fig. 2b). The projection of the object leaves two different regions divided by the contour around the object $\mathbf{C}$ (Fig. 2c): the foreground Ω_f (i.e. the pixels which lie inside the object) and the background Ω_b (i.e. the rest of the image). The contour $\mathbf{C}$ is a closed line which evolves through frames as the object moves in the scene. It is implicitly represented by the zero level-set of an embedding function $\Phi(\mathbf{x})$, i.e. $\mathbf{C} = \{\mathbf{x}|\Phi(\mathbf{x}) = 0\}$, where $\mathbf{x} = (x, y)$ is the location of a pixel in the image $\mathbf{I}$.

Each region has its own statistical appearance model, $P(\mathbf{y}|M_f)$ and $P(\mathbf{y}|M_b)$ respectively, being $\mathbf{y}$ the pixel value of the image ($\mathbf{I}(\mathbf{x}) = \mathbf{y}$) and $M = \{M_f, M_b\}$ the model parameter of the foreground or background. The appearance models can be obtained by



(a) (b) (c)

(d) (e) (f)

Fig. 2: (a) Example of 3D model. (b) Wireframe of the 3D model projected to the image. (c) The projection generates a contour **C** which separates the image domain in foreground (Ω_f) and background (Ω_b). (d) Proportionally highlighted in green, the likelihood of the pixels to belong to the foreground. (e) New frame introduced with the previous contour in blue. (f) After the optimization, the PWP3D algorithm is able to retrieve the updated pose (in red).

providing an initial bounding box of the object and building a histogram or probability density function describing the color distribution in the region. In Fig. 2d there is an example of the likelihood of every pixel of being foreground: the greener, the more likely to be foreground.

An energy function can then be formulated to maximize the discrimination between background and foreground. Instead of using the likelihoods, [4] proposed formulating this using the posterior in order to perform a pixel-wise marginalization of the model parameters. From [21], we have the energy function:

$$E(\Phi) = -\sum_{\mathbf{x}\in\Omega} \log\left(H_e(\Phi)P_f + (1-H_e(\Phi))\,P_b\right) \tag{1}$$

where $H_e(\Phi)$ is the smoothed Heaviside step function, which returns 1 inside the object and 0 in the outside. The function has been smoothed to provide uncertainty in the contours [4]. P_f and P_b are defined as:

$$P_f = \frac{P(\mathbf{y}|M_f)}{\eta_f P(\mathbf{y}|M_f) + \eta_b P(\mathbf{y}|Mb)} \tag{2a}$$

$$P_b = \frac{P(\mathbf{y}|M_b)}{\eta_f P(\mathbf{y}|M_f) + \eta_b P(\mathbf{y}|Mb)} \tag{2b}$$

where $\eta_f = \sum_{\mathbf{x}\in\Omega} H_e(\Phi)$ and $\eta_b = \sum_{\mathbf{x}\in\Omega} (1-H_e(\Phi))$. The pose parameters are the parameters we want to optimize, which are correlated with the level-set function Φ. To

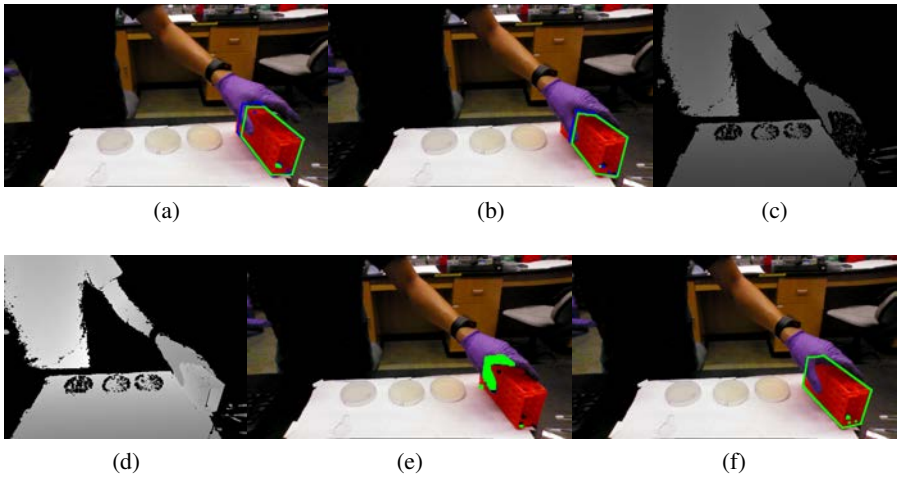


Fig. 3: (a) and (b): The pose of the tracked object is affected by the appearance of the occluding hand. New optimized pose is in green, and the pose from the previous frame is in blue. (c) Depth image from the case in (b). (d) As the pose is known from previous frames, the object can be rendered and merged with the depth image. (e) Occluding region Ω_o in green, i.e. the depth points over the object. (f) Establishing equal likelihoods in the Ω_o the algorithm is able to handle occlusions and obtain the correct pose.

solve this, equation (1) is differentiated with respect to the pose parameters to use standard gradient-based non-linear minimization techniques. Implementation details are in [21]. In consecutive frames, the pose from the previous frame is the initial solution to solve the optimization problem. In Fig. 2e there is the next frame of the example, where the most likely pixels that belong to the tracked object have been highlighted in green and the contour of previous iteration is drawn in blue. By solving the energy minimization problem we are able to obtain the new pose, as the red contours from Fig. 2f show.

The statistical appearance model needs to be updated online during the execution of the program. Otherwise, simple but common actions such as projecting shadow over the object may decrease the likelihood of the affected pixels to be considered as foreground, and therefore the performance of the algorithm may produce negative results. To do so, [21] propose the following adaptation equation:

$$P_t(\mathbf{y}|M_i) = (1-\alpha_i)P_{t-1}(\mathbf{y}|M_i) + \alpha_i P_t(\mathbf{y}|M_i), \quad i = \{f,b\} \tag{3}$$

where t and $t-1$ means the current and previous moment in time (frames), and α_f and α_b are the learning rate of the foreground and background respectively.

3.2 Handling occlusions with depth information

Certain situations may cause the PWP3D to produce undesirable results. For example, the algorithm is not able to handle occlusions properly. Until certain extent it does

overcome occluded situations, but often the grade of occlusion is too high or it lasts too long that the optimization tends to wrongly accommodate the object to maximize the discrepancy between foreground and background. Usually, when the occlusion is removed, the pose is corrected. However, for instance, if we want to perform some kind of online task supervision, those momentarily wrong poses defeat the purpose and are unacceptable.

Looking at the example in Fig. 3a we can see how when the hand (composed by likely background pixels) enters the foreground area (the blue contour from previous iteration) the PWP3D starts changing the pose to compensate for this. The optimized pose leaves a new contour (in green) which maximizes the discrepancy between foreground and background by leaving some red pixels out but letting some purple pixels in. This struggle of the PWP3D algorithm intensifies when the level of occlusions rises, such as a few frames later in Fig. 3b. The problem may be aggravated if the statistical appearance model of the foreground learns colors coming from the background.

To handle those occlusions is necessary to locate the occluding pixels and give them special treatment. Specific solutions to perform this operation can be found for every case: for example, a simple color image segmentation could be used to find the purple gloves in the scene from Fig. 3. However, we propose to use the information from the depth camera to serve at this purpose instead in order to provide a more general solution. The goal of this work is to track *difficult objects*, objects unable to reflect depth information, as it can be observed in the absence of depth data from the tube holder in Fig. 3c (black pixels mean no depth data available). From an initial known pose, obtained from the PWP3D or provided as input, we can use rendering software to get the z-buffer and merge it with the depth image if the camera system calibration is known (which we assume it is). A synthetic depth image with the 3D object seamlessly integrated can be obtained (Fig. 3d). Most importantly, we are able to retrieve the pixels from the depth camera which are in front of the object and therefore occluding it. An example of the detection of the occluding pixels is shown in Fig. 3e. Next, we describe the proposed algorithm in which this information is used:

We define a third type of region complementary to the background and the foreground: the occluding region Ω_o. Conceptually is like the intersection between Ω_f and Ω_b, since the pixel color information presents higher likelihood of it being background, but the 3D information reveals the possibility of part of the object being hidden behind. Thus, if we set the likelihood of the pixels in Ω_o to belong to the foreground and background as equal:

$$P(\mathbf{I}(\mathbf{x})|M_f) = P(\mathbf{I}(\mathbf{x})|M_b), \quad \forall \mathbf{x} \in \Omega_o \tag{4}$$

from (2) we have $P_f = P_b = P_o$ and, therefore, from (1)

$$E(\Phi) = -\sum_{\mathbf{x}\in\Omega_o} \log(P_o) = \mathrm{C}, \quad \forall \mathbf{x} \in \Omega_o \tag{5}$$

which means that, the pixels in Ω_o have no impact in the optimization process as their energy is constant in those points. That way, whenever an already correctly tracked object is occluded, the optimized pose is not affected by the occlusion, as we show in Fig. 3f.

During the optimization, every iteration the pose changes so the occluding region needs to be computed again. As we cannot ensure that the depth measurements are accurate enough, in a normal execution the occluding region is dilated several pixels to avoid missing pixels. Furthermore, when the foreground model is updated, those occluding pixels are removed so no wrong color information harms the model for succesive iterations. Although the inclusion of Ω_o as described assumes the occluding objects are observable with depth cameras, the idea can be generalized to any other type of occluding object detection. For example, if instead of using the depth straightforwardly we use a more complex type of tracking of the occluding object, such as [24], there would be less missing pixels and more accuracy in the Ω_o. Also, all objects involved can be considered by doing multi-object tracking with PWP3D itself [21]. In the case of controlled robotics environment, internal sensors may reveal the position of the different pieces.

4 Experiments

The dataset we use for this work consists of a series of recordings of biology students doing some procedures in a wet lab. The scenes are not arranged in any way, the people involved behave in a completely natural way. They have been recorded using three different RGB-D cameras from different perspectives. However, we only focus on the camera in front of the student, whose field of view covers the table where the activities are being performed. The RGB-D camera prototype used has a high resolution 1080p RGB camera, and a depth camera of 640×480 pixels. It is thought to be included as a default webcam in laptops, and has the advantage of providing depth measurements in a rather short range (20cm to 150cm) in contrast to other RGB-D models such as ASUS Xtion Pro or Microsoft Kinect (0.5m to 5m). Even in this short range the quality of the depth measurements is not good enough to locate and track some objects present in the scene, which are intrincate or transparent (e.g. test tubes, tube-holders, jars, flasks). On the other hand, depth information can still be used in our benefit, e.g. the table is clearly visible, as well as some other big parts of the scene, such as the body of the student. The color images have high resolution, which we resize to half their default size in order to reduce computational cost (960×540 pixels). The models of the objects have been created manually using open source 3D graphics software, as they were not part of any known dataset.

As the main goal of this work is to perform tracking and not detection, initially, an approximate pose of the object in the scene was provided by manually clicking several correspondences in the image and solving a PnP problem. For the tracking we used the open source implementation from [21] and made modifications over it, the main one being the introduction of the occluding region in the optimization problem. We keep the same learning rates for the foreground and background models ($\alpha_f = 0.05$, $\alpha_b = 0.02$) as it leads to good results in our experience. However, we observed an improvement in the accuracy of the pose when the band of pixels around the contour where the energy function is evaluated is fixed to 4 instead of 16. Due to the nature of the dataset, the displacements of the object between consecutive frames may be often very big. For this reason and our interest in accuracy and robustness over efficiency, in the gradient

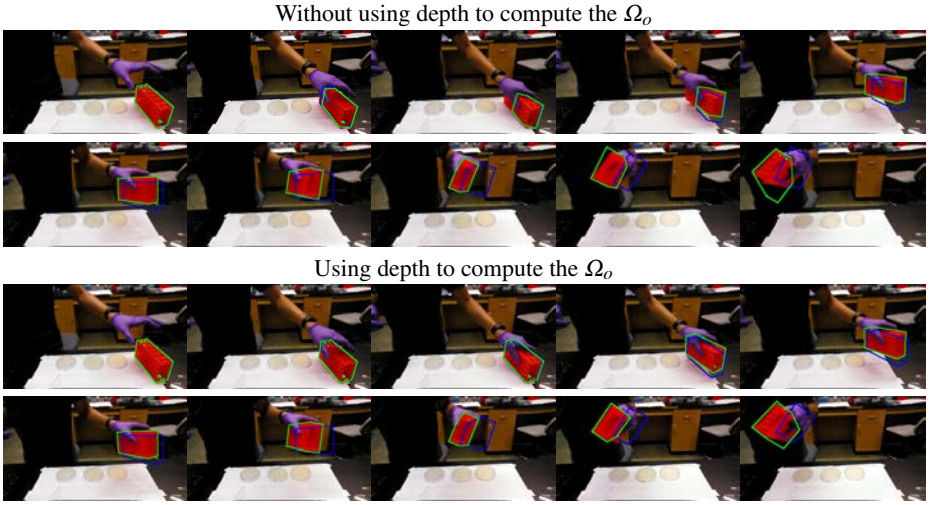


Fig. 4: Tracking sequence 1. The previous (blue) and current (green) contours are shown in each frame. The first two rows present the tracking without using depth information whereas the bottom rows use depth information, showing clear improvement.

descent optimization we had to keep small steps and increase the number of iterations up to 20 multi-iteration processes of 8 iterations each. After each multi-iteration, the occluding region is computed again with the current pose.

In Figures 4-6, we show some results of our method running the described dataset. In Fig. 4 there are some excerpts from the tracking sequence 1. To compare the performance when our enhancements are included, we show 10 frames without depth information and other 10 with depth (and therefore computation of the occluding region). In each frame the previous contour (pre-optimization) in blue and the current contour (post-optimization) in green are shown. When the hand is detected and removed from the optimization, it can be seen that the estimated poses are not deviated in the presence of occlusions. At the end the user moves too fast and gets blurrier and in both cases leads to inaccurate poses, aggravated when depth is not used. In Fig. 5 there is another example where the success of our contribution can be clearly observed. However, in both cases the pose is wrong when part of the object goes out of the image (not shown). In Fig. 6 we show an example where we can see how the depth points in the object do not reveal any relevant information. With our method, the depth points can be correctly computed and we can generate a synthetic 3D image with the tracked objects (Fig. 6), or insert the 3D model in the point cloud (Fig. 7).

5 Conclusion

In this work, we tackle the problem of object pose tracking with RGB-D sensors from a different perspective. We focus on those objects whose material or shape properties prevent the depth sensor to provide meaningful information. Starting from a state of the

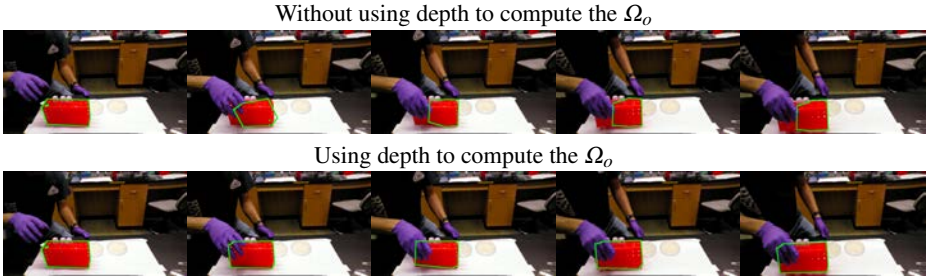


Fig. 5: Tracking sequence 2. The current contours are shown in green in each frame. The first row presents the tracking without using depth information whereas the bottom row uses depth information showing clear improvement.

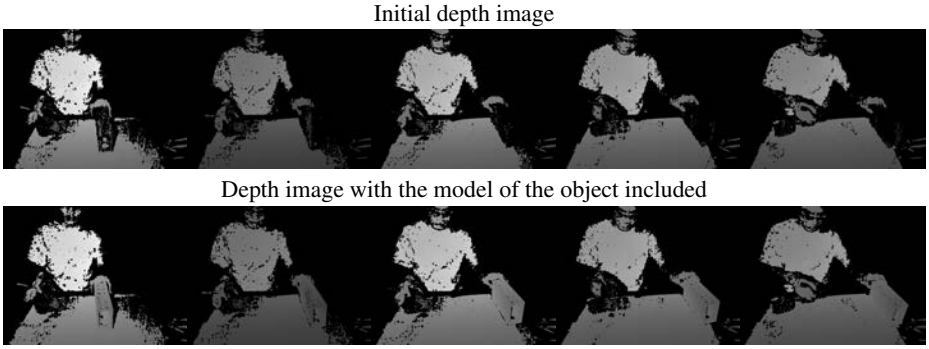
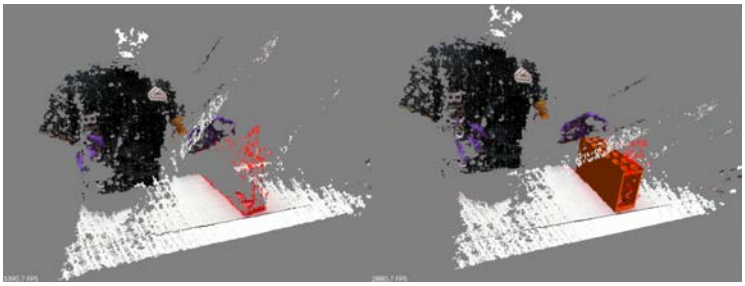


Fig. 6: Tracking sequence 3. Depth image of the sequence as provided by the system (first row). The object to track almost have no depth points during the sequence (black pixels are pixels without depth data). Being able to track the pose with the color camera we can insert the depth points of the object in the depth map (bottom row).

art color-based object tracking algorithm, we add some enhancements to improve the performance in our specific problem. Our main contribution is that we use the depth from the scene around the object to be tracked to detect occlusions. Here, we modify the optimization problem from the state of the art method to handle occlusions properly. Working on our own dataset, based on humans naturally manipulating objects in a wet laboratory, we show how our enhancements improve notably the performance of the system. The pose of the objects is maintained even under occluding situations, enabling these results to be used in higher level problems, such as activity recognition, online task supervision or robot manipulation.

Our method could be used standalone or in combination with other depth-based tracking algorithms which are unable to work with complex objects. Further enhancements can be done following the concept of using depth from the background when the objects have no depth information. For example, in a table-top configurations such as the one in our dataset, interpenetration of objects with the table should lead to detect



(a) (b)

Fig. 7: Point cloud snapshot as provided by the RGB-D sensor (a) and inserting the 3D model as proposed in this work (b). It can be observed in (a) that the points of the object (in red) are mostly out of place.

forbidden poses which would help the pose recovery process. Also, adding dynamics to the system could improve the optimization when, instead of just the information from the previous frame, a statistical estimation of the object pose is used.

Acknowledgments. This work was supported by Projects DPI2014-61792-EXP and DPI2015-65962-R (MINECO/FEDER, UE) and grant BES-2013-065834 (MINECO).

References

1. Asif, U., Bennamoun, M., Sohel, F.: Real-time pose estimation of rigid objects using RGB-D imagery. IEEE Conference on Industrial Electronics and Applications pp. 1692–1699 (2013)
2. Azad, P., Asfour, T., Dillmann, R.: Combining appearance-based and model-based methods for real-time object recognition and 6D localization. IEEE International Conference on Intelligent Robots and Systems pp. 5339–5344 (2006)
3. Azad, P., Munch, D., Asfour, T., Dillmann, R.: 6-DoF model-based tracking of arbitrarily shaped 3D objects. IEEE International Conference on Robotics and Automation pp. 5204–5209 (2011)
4. Bibby, C., Reid, I.: Robust Real-Time Visual Tracking using Pixel-Wise Posteriors. In: European Conference on Computer Vision. pp. 831–844. Springer (2008)
5. Byung-Kuk Seo, Hanhoon Park, Jong-Il Park, Hinterstoisser, S., Ilic, S.: Optimal Local Searching for Fast and Robust Textureless 3D Object Tracking in Highly Cluttered Backgrounds. IEEE Transactions on Visualization and Computer Graphics 20(1), 99–110 (2014)
6. Choi, C., Christensen, H.I.: Robust 3D visual tracking using particle filtering on the SE(3) group. IEEE International Conference on Robotics and Automation 31, 4384–4390 (2011)
7. Choi, C., Christensen, H.I.: 3D textureless object detection and tracking: An edge-based approach. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 3877–3884 (2012)
8. Dambreville, S., Sandhu, R., Yezzi, A., Tannenbaum, A.: Robust 3D Pose Estimation and Efficient 2D Region-Based Segmentation from a 3D Shape Prior. In: European Conference on Computer Vision. pp. 169–182. Springer (2008)

9. Drummond, T., Cipolla, R.: Real-time visual tracking of complex structures. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), 932–946 (2002)
10. Grundmann, T., Eidenberger, R., Schneider, M., Fiegert, M., Wichert, G.: Robust high precision 6D pose determination in complex environments for robotic manipulation. In: Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation at the IEEE International Conference of Robotics and Automation (2010)
11. Grundmann, T., Fiegert, M., Burgard, W.: Probabilistic Rule Set Joint State Update as approximation to the full joint state estimation applied to multi object scene analysis. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2047–2052 (2010)
12. Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., Lepetit, V.: Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In: IEEE International Conference on Computer Vision. pp. 858–865 (2011)
13. Imperoli, M., Pretto, A.: D^2CO: Fast and robust registration of 3d textureless objects using the directional chamfer distance. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9163, 316–328 (2015)
14. Kim, K., Lepetit, V., Woo, W.: Keyframe-based modeling and tracking of multiple 3D objects. In: IEEE International Symposium on Mixed and Augmented Reality. pp. 193–198. IEEE (2010)
15. Klank, U., Carton, D., Beetz, M.: Transparent object detection and reconstruction on a mobile platform. In: IEEE International Conference on Robotics and Automation. pp. 5971–5978 (2011)
16. Lepetit, V., Fua, P.: Monocular Model-Based 3D Tracking of Rigid Objects: A Survey. Now Publishers Inc. (2005)
17. Liu, M.Y., Tuzel, O., Veeraraghavan, A., Chellappa, R.: Fast directional chamfer matching. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 1696–1703 (2010)
18. Lowrey, K., Kolev, S., Tassa, Y., Erez, T., Todorov, E.: Physically-Consistent Sensor Fusion in Contact-Rich Behaviors. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 1656–1662 (2014)
19. Lysenkov, I., Eruhimov, V., Bradski, G.: Recognition and Pose Estimation of Rigid Transparent Objects with a Kinect Sensor. Robotics: Science and Systems (2012)
20. Phillips, C.J., Derpanis, K.G., Daniilidis, K.: A novel stereoscopic cue for figure-ground segregation of semi-transparent objects. In: IEEE International Conference on Computer Vision Workshops. vol. 1, pp. 1100–1107 (2011)
21. Prisacariu, V.A., Reid, I.D.: Pwp3d: Real-time segmentation and tracking of 3d objects. International Journal of Computer Vision 98(3), 335–354 (2012)
22. Rosenhahn, B., Brox, T., Weickert, J.: Three-Dimensional Shape Knowledge for Joint Image Segmentation and Pose Tracking. International Journal of Computer Vision 73(3), 243–262 (2007)
23. Schmidt, T., Hertkorn, K., Newcombe, R., Marton, Z., Suppa, M., Fox, D.: Depth-based tracking with physical constraints for robot manipulation. In: IEEE International Conference on Robotics and Automation. pp. 119–126 (2015)
24. Schmidt, T., Newcombe, R., Fox, D.: Dart: dense articulated real-time tracking with consumer depth cameras. Autonomous Robots 39(3), 239–258 (2015)
25. Schulman, J., Lee, A., Ho, J., Abbeel, P., Berkeley, U.C.: Tracking Deformable Objects with Point Clouds. In: IEEE International Conference on Robotics and Automation. pp. 1122–1129 (2013)

26. Ulrich, M., Wiedemann, C., Steger, C.: CAD-based recognition of 3D objects in monocular images. In: IEEE International Conference on Robotics and Automation. pp. 1191–1198 (2009)
27. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. Acm computing surveys (CSUR) 38(4) (2006)