Tracking Unobservable Rotations by Cue Integration

Ville Kyrki Lappeenranta University of Technology
Email: kyrki@lut.fi Danica Kragic Centre for Autonomous Systems
Royal Institute of Technology
Stockholm, Sweden
Email: danik@nada.kth.se

Abstract— **Model based object tracking has earned significant importance in areas such as augmented reality, surveillance, visual servoing, robotic object manipulation and grasping. Although an active research area, there are still few systems that perform robustly in realistic settings. The key problems to robust and precise object tracking are outliers caused by occlusion, self-occlusion, cluttered background, and reflections. Two most common solutions to the above problems have been the use of robust estimators and the integration of visual cues.**

The tracking system considered in this paper achieves robustness by integrating model-based and model-free cues. As model-based cues, we consider a CAD model of the object known a priori and as model-free cues, automatically generated corner features are used. The main idea is to account for relative object motion between consecutive frames using integration of the two cues. The particular contribution of this work is the integration framework where not only polyhedral objects are considered. In particular, we deal with spherical, cylindrical and conical objects for which the complete pose cannot be estimate using only CAD like models. Using the integration with the model-free features, we show how a full pose estimate can be obtained. Experimental evaluation demonstrates robust system performance in realistic settings with highly textured objects.

I. Introduction

Methods for real-time 3-D tracking of objects based on vision sensor data as well as the required hardware have recently been developed far enough to be applied in real-world applications. Possible tasks range from robot-control to augmented reality and medical applications [1], [2], [3], [4], [5], [6], [7]. Critical characteristics of such systems include their robustness to occlusion, image-noise and miss-tracking. Tracking based on a 3-D model of the target (e.g. [3]) provides robustness to some degree and avoids the drift in tracking. Another recently introduced method for increasing the robustness is the integration of model-free features together with model-based tracking [8].

Typically the object models used in 3-D tracking are edge-based, composed of line segments and planar surface patches. While these primitives are useful for many applications, some objects such as spheres are inconvenient to describe using them. Higher-order primitives, such as spheres, cylinders, and truncated cones, are still very useful models because they are efficient simplifications for many real-world objects.

A problem in the use of the higher order primitives is that if an object is described using a single primitive, such as a sphere, one or several degrees of freedom (dof) of the motion remains unobservable. For example, tracking a sphere using the edges leaves all rotational dofs unknown. In addition, even if CAD models have been effective for tracking due to their simple geometry, the problem of arbitrary textures and how to deal with these have not been thoroughly investigated.

As the main contribution of this paper, we present how the unobservable dofs can be observed in the model-based tracking by incorporating model-free features. The model-free features are automatically initialized and the two types of cues are integrated in a Kalman filter framework. The model-free measurements can be integrated as such, or they can be used to solve the full pose by optimization. The final system does not suffer from drift in any of the dofs which can be observed using the model, which greatly increases the stability of the tracking process.

In Section II some of the related work is reviewed. In Section III, the basic approach used for model based tracking is presented. Initialization of model-free features is described in Section IV, followed by the integration approach in Section V. Experimental evaluation is presented in Section VI and final discussion given in Section VII.

II. Related Work

3-D object tracking is typically performed by considering only object boundaries. In addition, most of the current systems concentrate on tracking of polyhedral or locally-planar objects. RAPID system, presented in [9], uses the dynamic vision approach presented by Dickmanns et al. [10], which is based on the use of extended Kalman filtering to integrate image measurements through a non-linear measurement function for pose estimation. A method that chains together edges, which are then matched and fitted to model, was presented by Lowe in [11]. An approach based on iterative minimization for finding the pose transformation that aligns a set of lines and ellipses, after which the pose is integrated in a linear Kalman filter over time, was presented in [12].

Recently, an approach for model based tracking related to bundle adjustment has been presented [13]. It relies on the use of a CAD model of the object and requires off-line matching of model points to their 2-D projections in a set of reference keyframes. Here, a keyframe methodology is applied to a set of local image patches in order to achieve viewpoint invariance by capturing the appearance of an object part in several views. Matching between the current and a keyframe is based on homographies, which is suitable for locally planar (polyhedral) objects. Compared to this approach, in our work we are interested in integrating CAD models with on-line generated surface features for tracking non-polyhedral objects. By generating the features on-line, we skip the off-line learning process required in the above work.

Work presented in [14] demonstrates a tracking system based on integration of visual and inertial sensors. A good performance for fast camera movements is achieved due to the integration with an inertial sensor but it is argued that, in order to have a robust system, more stable visual features (beacons) should be considered. We believe that our approach provides a suitable framework for retrieving such features.

Integration of vision based cues has been found to provide robustness in tracking and has been used successfully in many applications [15], [16]. Nevertheless, multiple cues have been applied mostly in 2-D tracking applications. Only recently they have been proposed for 3-D tracking [17], [8]. While the above demonstrate the integration only for case of polyhedral objects, we concentrate here on the problem formulation for solids of revolution and spherical objects.

III. Model-based tracking

Our approach is based on the tracking method presented in [3] which is uses a Lie algebra framework to estimate the six-dimensional pose of an object in a camera/world coordinate frame. Shortly, this is a full scale non-linear pose computation algorithms based on a 1D search along the projected edge normals in subsequent frames.

Let us now consider the case of an object with various 3-D features $\mathbf{S}$ where ${}^{o}\mathbf{S}$ represents the 3-D parameters of these features in the object frame. A camera reference frame is defined with its pose relative to the object frame defined by $\mathbf{r} = (\mathbf{t}, \Omega)^T$ representing the position and orientation of the camera relative to the object. Here, $\mathbf{t} = (t_x, t_y, t_z)$ is a vector of translation parameters along the x, y and z axes and $\Omega = (\Omega_x, \Omega_y, \Omega_z)$ is the rotation parameters around the x, y and z axes.

A. Tracking using a Lie group formulation

Here, the estimation of relative pose change between two consecutive frames, is based on the distance of sample point features generated in the image given the current pose estimate to the closest edge. This normal flow is then projected in the direction of the contour normal so as to represent an error we wish to minimize, see Fig. 1.

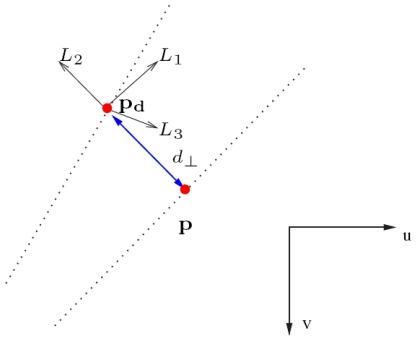

Fig. 1. Estimation of normal flow.

For the simple case of a point in 3-D, a projection matrix is defined as $\mathbf{P} = \mathbf{KM}$ in the coordinate system of the structure, where $\mathbf{K}$ is composed of the intrinsic camera parameters and $\mathbf{M}$ is composed of the extrinsic pose parameters.

The projective coordinates of a point are then given by:

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \mathbf{P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{1}$$

where the 2-D point $\mathbf{p} = (\mathbf{x}, \mathbf{y})$ of the 3-D point $\mathbf{P}$ in image pixel coordinates coordinates is given by $x = (u/w)$ and $y = (v/w)$

To estimate the relative change in pose between two consecutive frames, the derivative of $SE(3)$ corresponding to the tangent velocity vector space or Lie Algebra is used. Here, velocity basis matrices, called generators, are chosen in a standard way to represent translations in the x, y and z directions along with rotations around the x, y and z axes. These six generators are given as:

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{G}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2}$$

$$\mathbf{G}_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{G}_6 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

These generators form a basis for the vector space of derivatives of $SE(3)$ at the identity. The Lie Group of displacements is related to the Lie Algebra of velocities via the exponential map, so that:

$$\mathbf{A} = \exp(\mathcal{A}) = \exp(\sum \alpha_i \mathbf{G}_i) \tag{3}$$

where each α_i corresponds to an element of the kinematic screw or twist representing the inter-frame transformations.

The motion in the image is related to the twist in 3-D by taking the partial derivative of projective image coordinates with respect to the i^{th} generating motion:

$$\begin{pmatrix} u' \\ v' \\ w' \end{pmatrix} = \mathbf{PG}_i \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4}$$

with the motion in pixels being equivalent to the well known optical flow equation as:

$$\mathbf{L}_i = \begin{pmatrix} \tilde{u}' \\ \tilde{v}' \end{pmatrix} = \begin{pmatrix} \frac{u'}{w} - \frac{uw'}{w^2} \\ \frac{v}{w} - \frac{vw}{w^2} \end{pmatrix} \tag{5}$$

Continuing with the determination of the inter-frame movement, the different generating motions are projected in the direction of the normal of the contour as:

$$f_i = L_i . \hat{n} \tag{6}$$

where $\hat{n}$ is the normal direction.

B. Minimization

Computing the motion is performed by solving a weighted least-squares algorithm as follows:

$$v_i = \sum_{\xi} s(d^{\xi}) d^{\xi} f_i^{\xi}, \tag{7}$$

$$C_{ij} = \sum_{\xi} s(d^{\xi}) f_i^{\xi} f_j^{\xi}, \tag{8}$$

$$\alpha_i = \sum_{j} C_{ij}^{-1} v_j, \tag{9}$$

where d^{ξ} is a distance measured normal to a contour in the image and α_i is an estimated velocity corresponding to one of the six basis generators i. Here, the f_i represent elements of the interaction matrix for a distance to a point and $s(d^{\xi})$ is a robust weighting function.

Rodriguez's formula is then used for the exponential mapping of the velocity parameters to the corresponding instantaneous displacement and hence pose update. To apply the update to the displacement between the object and camera the exponential map is applied using homogeneous matrices resulting in:

$${}^{c}\mathbf{M}_{o,t+1} = \mathbf{M}_{o,t} \quad \exp\left(\sum_i \alpha_i \quad \mathbf{G}_i\right) \tag{10}$$

C. Unobservable degrees of freedom

As mentioned previously, we concentrate mainly on model based tracking of solids of revolution and spherical objects. Solids of revolution, such as cylinders and truncated cones, have one unobservable degree of freedom, namely the axis of revolution. With an assumption that Lie generators, (2), are aligned with the coordinate system attached to the object, the effect of the generator corresponding to this rotation can be removed (in our case G_5 as the axis of revolution corresponds to y). We note here that the expressions (9)-(10) only lose one dimension of i, and otherwise the estimation of the object pose remains the same. A spherical object has three unobservable degrees of freedom, as no rotations can be detected. In this case, all three generators related to the rotational motion (G_4, G_5, G_6) are disregarded.

IV. Initialization of model-free features

For tracking of model-free features, we have adopted the approach suggested in [8], which uses Harris corner detector for point initialization and SSD (sum of squared differences) for tracking. Here, we will only present the modifications to the point initialization due to the more complex geometric primitives used.

When a new model-free point is initialized, its 3-D location in the object coordinate frame is recorded. If the point lies on a planar patch, determining the location corresponds to calculating the intersection of the plane and the line corresponding to the point where the 3-D point projects onto the camera image plane. In this case the solution is unique. Contrary to this, if the object model is a sphere or a truncated cone, there may be several intersections.

Consider the sphere in Fig. 2. Let r be the radius of a sphere, $\mathbf{t}$ the location of its origin (located at its center), $\mathbf{R}$ the rotation matrix corresponding to its current orientation, and $\mathbf{p}$ be the homogeneous coordinates of the point in the camera frame, that is, the intrinsic calibration of the camera has been taken into account. Then, the intersection of the sphere and the image ray is of form $k\mathbf{p}$ where k is a solution of

$$(kp_x - t_x)^2 + (kp_y - t_y)^2 + (kp_z - t_z)^2 - r^2 = 0. \tag{11}$$

When the ray passes through the sphere, there are two solutions, corresponding to the two intersections. The solutions are

$$\begin{aligned} k_1 &= \frac{\mathbf{p}\cdot\mathbf{t} + \sqrt{(\mathbf{p}\cdot\mathbf{t})^2 - \|\mathbf{p}\|^2(\|\mathbf{t}\|^2 - r^2)}}{\|\mathbf{p}\|^2} \\ k_2 &= \frac{\mathbf{p}\cdot\mathbf{t} - \sqrt{(\mathbf{p}\cdot\mathbf{t})^2 - \|\mathbf{p}\|^2(\|\mathbf{t}\|^2 - r^2)}}{\|\mathbf{p}\|^2} \end{aligned} \tag{12}$$

and the smaller one, k_2, corresponds to the desired closest intersection. Finally, the intersection in object frame is

$$\mathbf{q} = \mathbf{R}^T(k_2\mathbf{p} - \mathbf{t}). \tag{13}$$

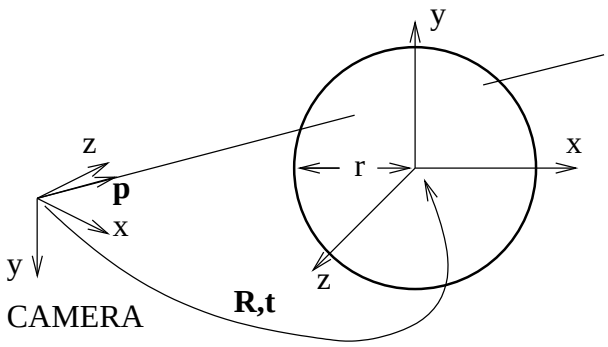

Fig. 2. Sphere model.

For a truncated cone, let r_b and r_t be the radiuses of the bottom and top, correspondingly, and let h be the height (see Fig. 3). Note that this model applies also to cylinders, for which $r_b = r_t$. Let the origin of the object frame be at the center of the bottom, and $\mathbf{t}$ and $\mathbf{R}$ describe its position and orientation. The line corresponding to the image point $\mathbf{p}$ can then be written in object frame as $\mathbf{l} = -\mathbf{R}^T\mathbf{t} + k\mathbf{R}^T\mathbf{p} \equiv \mathbf{w} + k\mathbf{v}$. Defining $a = \frac{r_t - r_b}{h}$ as the unit change of the radius, and fixing the y-axis of the object coordinate system to the axis of revolution, the intersection can be found from

$$l_x^2 + l_z^2 = (r_b + a l_y)^2, \tag{14}$$

which results in

$$\begin{aligned} k^2(v_x^2 + v_z^2 - a^2 v_y^2) + 2k(v_x w_x + v_z w_z - a^2 v_y w_y - 2a r_b v_y) \\ + w_x^2 + w_z^2 - a^2 w_y^2 - 2a r_b w_y - r_b^2 = 0. \end{aligned} \tag{15}$$

This second order polynomial has again up to two solutions, k_1, k_2, giving the intersections $-\mathbf{R}^T\mathbf{t} + k_i\mathbf{R}^T\mathbf{p}$. Now the intersections are directly expressed in the object frame as desired, but to select the closest one to the camera, the intersections must be transformed to camera frame, and then the closest one can be chosen.

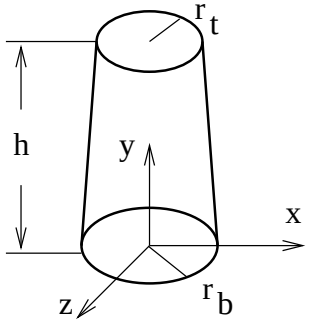

Fig. 3. Cone model.

V. Integration

The information from the two types of cues are integrated in an iterated extended Kalman filter (IEKF) framework. The system state $\mathbf{x}$, describing the 6 dofs of SE(3), is modeled as

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i + \mathbf{w}_i \\ \mathbf{y}_i &= h(\mathbf{x}_i) + \mathbf{v}_i \end{aligned} \tag{16}$$

where $\mathbf{y}$ is the measurement, $h(\cdot)$ is the measurement model, and $\mathbf{w}_i$ and $\mathbf{v}_i$ are zero-mean Gaussian sequences describing model and measurement errors. Additional system parameters include the covariances of the system state $\mathbf{P}_i = E[\mathbf{x}_i\mathbf{x}_i^T]$, model error $\mathbf{Q}_i = E[\mathbf{w}_i\mathbf{w}_i^T]$, and measurement error $\mathbf{S}_i = E[\mathbf{v}_i\mathbf{v}_i^T]$.

In a Kalman filter, the state estimation is a two-step process: First the state is predicted, then updated according to new measurements. In our case the system prediction step does not change the state (see Eq. 16), but the state covariance is updated, corresponding to the growing uncertainty in the estimation as time passes.

To avoid the pitfalls due to discontinuities and non-uniqueness of three angle representations of rotation, we have followed the approach of Welch and Bishop [18] to represent the accumulated rotation outside the state vector $\mathbf{x} = (X, Y, Z, \phi, \theta, \psi)^T$ so that the angles ϕ, θ, ψ only represent incremental rotations around the coordinate axes, and the accumulated rotation is stored in matrix $\mathbf{R}_0$. Thus, after each measurement update of the Kalman filter, the angles are used as immediate angles to update $\mathbf{R}_0$, after which the angles in the state vector are set to zero.

We will present first how the symmetric axes have to be taken into account in the measurement model of the Kalman filter for model-based cues. After that, two different approaches for the integration of model-free cues are described.

A. Measurements with unobservable DOF

The measurement model $h(\cdot)$ describes how the measurements are related to the system state. In the case when all dofs are observable using the model, the measurement model for the model-based tracking can be written simply as

$$h_F(\mathbf{x}) = \mathbf{x} = \begin{pmatrix} X & Y & Z & \phi & \theta & \psi \end{pmatrix}^T. \tag{17}$$

It should be noted that in (17) and all following measurement models, the additive measurement error $\mathbf{v}$ has been left out to make the expression more readable, but it is assumed to be present as in (16). To convert the measured translation $\hat{\mathbf{t}}$ and rotation $\hat{\mathbf{R}}$ into the same form, function $g_F(\hat{\mathbf{t}}, \hat{\mathbf{R}})$ can be defined as

$$g_F(\hat{\mathbf{t}}, \hat{\mathbf{R}}) = \begin{pmatrix} \hat{\mathbf{t}} \\ \boldsymbol{\alpha}(\hat{\mathbf{R}}\mathbf{R}_0^T) \end{pmatrix}. \tag{18}$$

where $\boldsymbol{\alpha}(\mathbf{R})$ is a function making the conversion from matrix representation of rotation to immediate angles. Using this formulation, the gradient of the measurement function, needed to compute the Kalman gain, becomes simply an identity matrix:

$$\frac{\partial h_F(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{I}_6. \tag{19}$$

When all rotational dofs are unobservable using the model, they can be ignored, and the measurement functions become

$$h_S(\mathbf{x}) = \begin{pmatrix} X & Y & Z \end{pmatrix}^T \qquad g_S(\hat{\mathbf{t}}) = \hat{\mathbf{t}} \tag{20}$$

and the gradient is

$$\frac{\partial h_S(\mathbf{x})}{\partial \mathbf{x}} = \begin{pmatrix} \mathbf{I}_3 & \mathbf{0}_3 \end{pmatrix}. \tag{21}$$

When there is only one unobservable dof, such as in the case of a truncated cone, the conversion functions become more complex. The measurement function for the cone model presented in Sec. IV is

$$h_C(\mathbf{x}) = \begin{pmatrix} X & Y & Z & \alpha_x(\mathbf{R}_0^T\mathbf{R}_p) & \alpha_z(\mathbf{R}_0^T\mathbf{R}_p) \end{pmatrix}^T \tag{22}$$

where $\mathbf{R}_p$ is the predicted rotation. Thus, only rotations around x and z axes of the object frame can be measured. These angles are also represented as instantaneous, that is, as incremental angles compared to the previous time instant. It should be also noticed that, because the model has no predictive function, $\mathbf{R}_p = \mathbf{R}_0$, both α_x and α_z are zero. While only two dofs of rotational motion are measured, these two do not necessarily correspond to the rotations around particular two coordinate axes of the camera frame, but instead to some combination of the three axes. As a Kalman filter is a linearized model of the system, this is taken into account by linearization. In addition, the orientation of the object must be transformed from the camera frame to object frame. Therefore, the conversion function for the measurement is

$$g_C(\mathbf{x}) = \begin{pmatrix} \hat{\mathbf{t}} \\ \alpha_x(\mathbf{R}_0^T\hat{\mathbf{R}}) \\ \alpha_z(\mathbf{R}_0^T\hat{\mathbf{R}}) \end{pmatrix}. \tag{23}$$

While the absence of prediction caused both of the measured angles to be zero, the form of the function in (22) is still necessary for the calculation of the measurement gradient. The

gradient of the measurement function at $\mathbf{R}_p = \mathbf{R}_0$ becomes then

$$\frac{\partial h_C(\mathbf{x})}{\partial \mathbf{x}} = \begin{pmatrix} \mathbf{I}_3 & & \mathbf{0}_3 & \\ \mathbf{0} & R_p(1,1) & R_p(2,1) & R_p(3,1) \\ \mathbf{0} & R_p(1,3) & R_p(2,3) & R_p(3,3) \end{pmatrix}, \quad (24)$$

where $R_p(i,k)$ is the (i,k) element of $\mathbf{R}_p$.

B. Integration of model-free cues

For the integration of point measurements, two approaches are considered: 1) directly integrating the point measurements in the Kalman filter using a non-linear measurement model corresponding to the perspective projection, and 2) using M-estimators to robustly estimate the current pose and integrating the pose measurement using a linear measurement model. The two approaches have different computational complexities and error characteristics. It should be noted that in both approaches the following simplification is made compared to full structure-from-motion: For all point features, the 3-D object frame location is recorded during the initialization using the current pose estimate, as presented in Sec. IV. Therefore, the point location is not included in the Kalman filter state, but the associated uncertainty is modeled as part of the measurement uncertainty. This choice makes real-time operation possible even with a large number of points.

1) Direct integration: In the direct integration approach, the measurement function is the perspective projection of a known 3-D point $\mathbf{q}_i$:

$$\begin{aligned} \begin{pmatrix} X_j & Y_j & Z_j \end{pmatrix}^T &= \mathbf{R}_p(\mathbf{x}, \mathbf{R}_0)\mathbf{q}_j + \mathbf{t}(\mathbf{x}) \\ h_j(\mathbf{x}) &= \begin{pmatrix} X_j/Z_j & Y_j/Z_j \end{pmatrix}^T \end{aligned} \quad (25)$$

where $\mathbf{R}_p(\mathbf{x}, \mathbf{R}_0)$ is the rotation matrix taking into account the accumulated rotation $\mathbf{R}_0$ and the rotational dofs of the state $\mathbf{x}$, and $\mathbf{t}(\mathbf{x})$ represents the translational dofs of the state. The same function is used for each of the measured points. The gradient of the measurement function can be calculated from (25), but is not shown here for the sake of brevity.

The measurement errors are assumed to be independent with respect to the coordinate axes. They are also assumed to be independent for each point. Thus, the measurement covariance matrix for point measurements can be written as $\mathbf{S}_I = \sigma_i^2 \mathbf{I}$ where σ_i^2 is the image measurement variance.

2) M-estimators: The direct integration approach above suffers from measurement outliers. To increase the outliers tolerance, a robust M-estimator can be used [19]. To outline the approach, the M-estimator is first used to find the best pose corresponding to current measurements. Then, the model-free pose measurement is integrated in the Kalman filter with the model-based measurement.

The M-estimators are used to minimize a robust error function, based on the sum of the squared errors between image measurements and the recorded 3-D points. The sum of squared errors measure of the 3D-2D projection error can be written as

$$d = \sum_j e_j^2 \qquad e_j = \sqrt{\left(x_j - \frac{X_j}{Z_j}\right)^2 + \left(y_j - \frac{Y_j}{Z_j}\right)^2} \quad (26)$$

where (x_j, y_j) is the measured position, and (X_j, Y_j, Z_j) are its coordinates in the camera frame, from (25).

Instead of solving the optimization problem for the squared error criterion in (26), the squared residual is replaced with a robust weighting function. A Tukey weight function [20] is used and thus the optimization problem can be written

$$\min_{\mathbf{R},\mathbf{t}} \sum_j w_{TUK}(e_j) e_j^2 \quad (27)$$

with the weight function defined as

$$w_{TUK}(x) = \begin{cases} \left(1 - (x/c)^2\right)^2 & \text{if } x \le c \\ 0 & \text{if } x > c \end{cases} \quad (28)$$

where c is a parameter describing the outliers rejection threshold. The problem is solved as iteratively reweighted least squares, such that for each measurement, a weight is first determined, then a weighted least squares optimization is performed, and these two steps are repeated until convergence. The least squares optimization is performed by a Polak-Ribiere variant of the conjugate gradient method [21]. Each line search along the gradient is performed by first bracketing the minimum by golden section bracketing, and then locating it by Brent's method [22].

The estimated pose $(\mathbf{R}, \mathbf{t})$ is finally integrated in the Kalman filter using the measurement function identical to that of model-based tracking without unobservable dofs presented in (17). The conversion function is also identical to (18). The measurement uncertainty can be written as $\mathbf{S}_M = \begin{pmatrix} \sigma_t^2 \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \sigma_\phi^2 \mathbf{I}_3 \end{pmatrix}$. By adjusting the translational and rotational uncertainties σ_t and σ_ϕ, the effect of model-free measurements can be controlled. For example, using a very large σ_t makes the effect of model-free measurements negligible with respect to the translation of the object, such that only rotational dofs are estimated using them.

VI. Experimental evaluation

Experiments were performed to inspect both the applicability of the proposed methods, and to inspect the computational feasibility. Two test sequences are presented in Figs. 4 and 5, one for tracking a truncated cone and another for a sphere. The sequences are overlayed with the projected model edges, using the M-estimator approach for integration.

Figures 6 and 7 present the tracked angles for the sphere model undergoing the motion shown in Fig. 5 for both M-estimators and direct integration. The three curves correspond to the rotations around the three principal axes of the camera frame. For that reason, the rotation is not exactly around the vertical axis of the camera during the beginning of the scene, where the object is rotated around the world vertical axis. The rotation reaches its maximum near the 400th frame, corresponding to the fourth image in the first row of Fig. 5. After that the object is rotated backwards until frame 700, after which the object is moved and rotated along a more complex trajectory.

The graph in Fig. 6 represents the true motion relatively well while the direct integration approach presented in Fig. 7 detects rotation around z-axis which is not present. This

Fig. 4. Cone test sequence.

Fig. 5. Sphere test sequence.

demonstrates qualitatively that the M-estimator approach is able to detect the rotation even while the orientation is unobservable from the model point of view. One particular thing to notice is that the distance to the object is somewhat overestimated for a short while during the tracking (see the fourth image in Fig. 5). The distance is mostly estimated by the model-based tracker, and at this point one of the object edges has a gradual shadow and the edge cannot be reliably detected, which causes the overestimation of the distance. An important point to notice is that this does not seem to cause significant problems for the estimation of the orientation, as can be seen from the symmetry of the rotation trajectories in Fig. 6. In contrast, there are spurious changes present in Fig. 7, which are result from the bad quality of some of the tracked points. This sequence is used here to demonstrate that the M-estimators are important in reducing the effect of outliers, especially when the texture of the object does not present easy tracking targets.

Figures 8 and 9 present the tracked angles for the cone model under the sequence shown in Fig. 4 for both integration approaches. Consistent behavior can be seen in this sequence for both of the methods. In this sequence, the average number of tracked points is larger, and there are no spurious errors in the points. It should be noticed that the model-based tracker has here a significant contribution on the observable rotational dofs because the model-free features are initialized on-line and thus measure only relative changes, and for that reason they do not suppress consistently incorrect estimates of the model-based tracker.

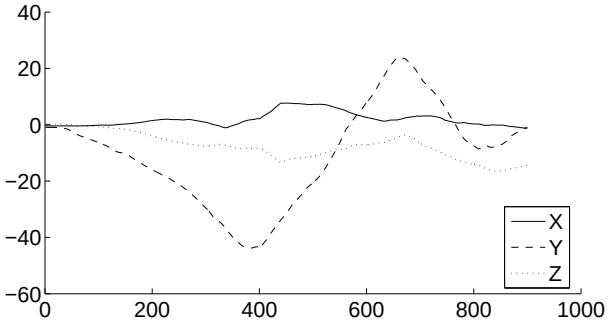

Fig. 6. Estimated sphere orientation with M-estimators.

The computational feasibility of the methods was examined by measuring the average estimation times over 1000 frames long sequences of images. The average per frame processing times are presented in Table I. The time required for retrieving the images is not included in the processing time. With the sphere sequence, tracking is slightly faster using the direct integration model, because the number of image features is small, keeping also the measurement of the Kalman filter low-dimensional. When this dimensionality grows, the processing time increases. With the cone sequence, the M-estimator approach has lower processing time compared to the direct integration, as its computational complexity grows slower with respect to the number of measurements. However, both approaches seem to be feasible for real-time use from the computational point of view.

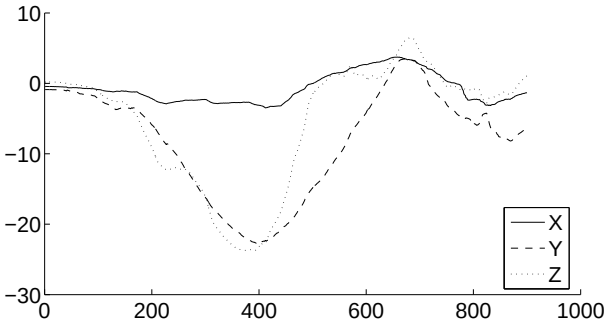

Fig. 7. Estimated sphere orientation with direct integration.

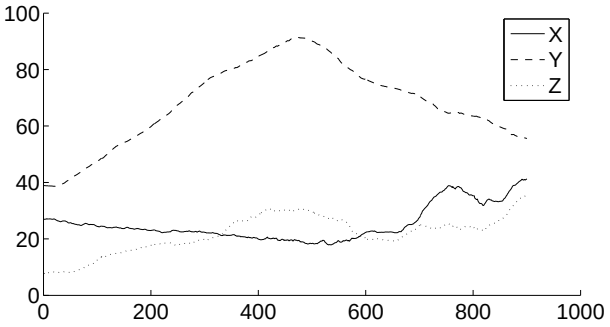

Fig. 8. Estimated cone orientation with M-estimators.

VII. Discussion and conclusion

In this paper, we have presented how model-free features can be used in conjunction with model-based tracking to observe the dofs which are unobservable from the model. During the initialization step, a reference pose is established for the unobservable dofs and all further measurements are with respect to this pose. We have demonstrated the idea on two types of geometric primitives, cones and spheres, corresponding to one and three unobservable dof.

Only qualitative assessment of the accuracy was done, as no ground truth data were available. Also, the method does not try to "close the loop", that is, lost points are not memorized for possible future use. This causes drift over time in the unobservable dofs. However, there is no drift in the observable dofs, which is a significant advantage that results in increased robustness.

In this work, we have presented a system which is able to track higher order geometric primitives with textured surfaces. We believe that such objects are important for augmented reality and robotic grasping purposes since many of the objects humans deal with (for example food or items of furniture) have complex texture and can be represented at least approximately as simple geometric primitives.

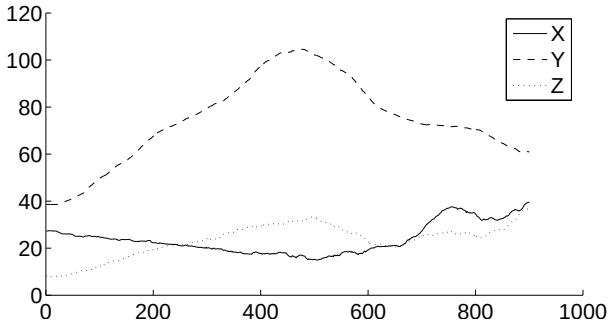

Fig. 9. Estimated cone orientation with M-estimators.

TABLE I

Average processing times

Method	Sphere	Cone
Direct integration	2.8 ms	5.4 ms
M-estimators	3.1 ms	5.0 ms

References

[1] M. Armstrong and A. Zisserman, "Robust object tracking," in *Proceedings of the Asian Conference on Computer Vision*, vol. I, pp. 58–61, 1995.

[2] D. Koller, K. Daniilidis, and H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *International Journal of Computer Vision*, vol. 10, no. 3, pp. 257–281, 1993.

[3] T. Drummond and R. Cipolla, "Real-time tracking of multiple articulated structures in multiple views," in *Proceedings of the 6th European Conference on Computer Vision, ECCV'00*, vol. 2, pp. 20–36, 2000.

[4] D. Lowe, *Perceptual Organisation and Visual Recognition*. Robotics: Vision, Manipulation and Sensors, Dordrecht, NL: Kluwer Academic Publishers, 1985. ISBN 0-89838-172-X.

[5] P. Wunsch and G. Hirzinger, "Real-time visual tracking of 3D objects with dynamic handling of occlusion," in *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'97*, vol. 2, pp. 2868–2873, 1997.

[6] M. Vincze, M. Ayromlou, and W. Kubinger, "Improving the robustness of image-based tracking to control 3D robot motions," in *Proceedings of the International Conference on Image Analysis and Processing*, pp. 274–279, 1999.

[7] E. Marchand and F. Chaumette, "Feature tracking for visual servoing purposes," *Robotics and Autonomous Systems*, vol. 52, no. 1, pp. 53–70, 2005.

[8] V. Kyrki and D. Kragic, "Integration of model-based and model-free cues for visual object tracking in 3d," in *IEEE International Conference on Robotics and Automation, ICRA'05*, pp. 1566–1572, 2005.

[9] C. Harris, "Tracking with rigid models," in *Active Vision* (A. Blake and A. Yuille, eds.), ch. 4, pp. 59–73, MIT Press, 1992.

[10] E. D. Dickmanns and V. Graefe, "Dynamic monocular machine vision," *Machine Vision and Applications*, vol. 1, pp. 223–240, 1988.

[11] D. G. Lowe, "Robust model-based motion tracking through the integration of search and estimation," *Int. J. of Comp. Vis.*, vol. 8, no. 2, pp. 113–122, 1992.

[12] P. Wunsch and G. Hirzinger, "Real-time visual tracking of 3-d objects with dynamic handling of occlusion," in *IEEE Int. Conf. on Robotics and Automation, ICRA'97*, (Albuquerque, New Mexico, USA), pp. 2868–2873, Apr. 1997.

[13] L. Vacchetti, V. Lepetit, and P. Fua, "Stable real-time 3d tracking using online and offline information," *IEEE Trans. on Patt. Anal. and Machine Intell.*, vol. 26, no. 10, pp. 1385–1391, 2004.

[14] G. Klein and T. Drummond, "Robust visual tracking for non-instrumented augmented reality," in *In Proc. 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 113–122, 2003.

[15] C. Rasmussen and G. Hager, "Probabilistic data association methods for tracking complex visual objects," *IEEE Trans. PAMI*, vol. 23, no. 6, pp. 560–576, 2001.

[16] D. Kragic and H. I. Christensen, "Cue integration for visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 18–27, Feb. 2001.

[17] G. Taylor and L. Kleeman, "Fusion of multimodal visual cues for model-based object tracking," in *Australiasian Conf. on Robotics and Automation*, (Brisbane, Australia), 2003.

[18] G. Welch and G. Bishop, "SCAAT: Incremental tracking with incomplete information," in *Proc. Computer graphics and interactive techniques*, (Los Angeles, CA, USA), pp. 333–344, August 3–8 1997.

[19] P. J. Huber, "Robust estimation of a location parameter," *Annals of Mathematical Statistics*, vol. 35, pp. 73–101, 1964.
[20] P. J. Huber, *Robust Statistics*. Wiley, 1981.
[21] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C++*. Cambridge University Press, 2002.
[22] R. P. Brent, *Algorithms for Minimization without Derivatives*. Prentice-Hall, 1973.