Articulated-body Tracking through Anisotropic Edge Detection

David Knossow, Joost van de Weijer, Radu Horaud, and Rémi Ronfard

INRIA Rhône-Alpes,
655 Av. de l'Europe, 38330 Montbonnot, France
{firstname.lastname}@inrialpes.fr

Abstract. This paper addresses the problem of articulated motion tracking from image sequences. We describe a method that relies on both an explicit parameterization of the extremal contours and on the prediction of the human boundary edges in the image. We combine extremal contour prediction and edge detection in a non linear minimization process. The error function that measures the discrepancy between observed image edges and predicted model contours is minimized using an analytical expression of the Jacobian that maps joint velocities onto extremal contour velocities. In practice, we model people both by their geometry (truncated elliptic cones) and their articulated structure – a kinematic model with 40 rotational degrees of freedom. To overcome the flaws of standard edge detection, we introduce a model-based anisotropic Gaussian filter. The parameters of the anisotropic Gaussian are automatically derived from the kinematic model through the prediction of the extremal contours. The theory is validated by performing full body motion capture from six synchronized video sequences at 30 fps without markers.

1 Introduction and background

In this paper, we address the problem of tracking complex articulated motions, such as human motion, from multiple camera image sequences using a solely contour-based approach. Articulated motion tracking has been thoroughly studied in the past few years using either one or multiple cameras and with or without artificial markers. Monocular approaches generally require a probabilistic framework such as in [1,2,3] to cite just a few. It requires prior knowledge: the mapping between articulated-motion space and image-data space must be learnt prior to tracking. However, learning the entire motion space of a 40 dof kinematic chain remains an open issue. Other authors have tried to recover articulated motion from image cues such as optical flow through sophisticated non-linear minimization methods [4,5].

To overcome the limitations of monocular approaches, methods based on multiple cameras have been proposed in the literature. These approaches generally use either image edges or silhouettes [6,7,8,9]. Furthermore, these methods use generic models, such as superquadrics, quadrics or simple cylinders, to represent body parts. Nevertheless, projecting these models onto images and comparing them with contours and/or

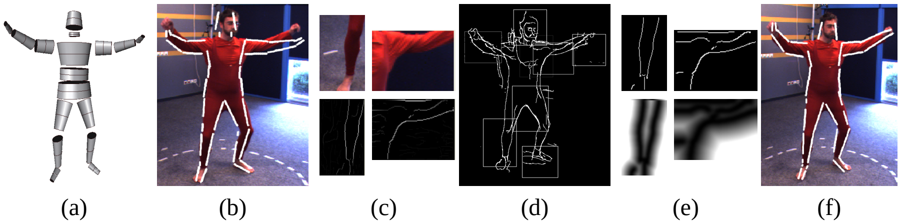


Fig. 1. From left to right: The current model (a) is matched against a set of new images (only one is shown here) (b). The contours in these images (c) and(d) are extracted using an anisotropic color Canny filter (c). They are compared with the predicted model contours using the Chamfer distance (e). Finally, the estimated model is consistent with the new image (f).

silhouettes is not an obvious task. In the case of sharp edges (surface discontinuities) there are well documented methods allowing for an explicit (analytic) representation of the mapping between the object's constrained (articulated) motion parameters and the observed image contours [10,11]. However, in the case of human motion tracking, the task is made much harder by the fact that the human body has few (if any) sharp edges and its silhouette stems from the projection of smooth surfaces rather than surfaces with sharp edges. Moreover, the silhouettes used by these methods are often unreliable due to background substraction problems around moving objects. Due to the lack of robustness of silhouette extraction we propose an approach that solely relies on contours.

For a contour-based approach to be successful the correct detection of object contours in the image is essential. Our approach to improve this contour detection consists of three steps. Firstly, we model articulated objects such as humans using smooth surfaces, namely truncated elliptic cones, as basic primitives which are joined together to form an articulated structure (Fig. 1.a). Each joint has one, two or three rotational degrees of freedom. This model allows us to explicitly parametrize the *extremal contours* of the model, which are the projection of the smooth surfaces onto the image (see Fig.2), in terms of the articulated structure parameters.

Secondly, we exploit the information provided by the kinematic model to perform a model-based edge detection. Well known methods such as Canny-Deriche [12], measure the first-order derivatives in an image. Convolution with Gaussian derivative filters make the measurement of image derivatives more robust. But the isotropic Gaussian filtering suffers from a blurring effect. Furthermore, crossing edges are not well detected. To overcome those flaws, the anisotropic Gaussian filter had been introduced in [13] and a fast implementation is proposed in [14]. Anisotropic Gaussian filtering smoothes image intensities along the predicted contour directions and computes directional derivatives across the predicted contours which improves robustness of the tracker [15]. Furthermore, to ensure optimal use of the available contrast in the image, color derivatives are applied as proposed in [16]. We combine both anisotropic filtering and color derivatives (Fig. 1- c,d) to obtain an anisotropic color Canny filter, to arrive at a final binary edge map (Fig. 1-e).

Finally, the tracking is performed by minimizing a distance between the predicted extremal contours and the observed contours. The process consists in minimizing an error function:

$$\min_{\boldsymbol{\Phi}} \; E(\mathcal{Y}, \mathcal{X}(\boldsymbol{\Phi})), \tag{1}$$

where E is a distance function, $\boldsymbol{\Phi} = (\phi_1, \ldots, \phi_p)$ is the n-dimensional vector whose components are the motion parameters, $\mathcal{Y}$ is the set of observed image contours and $\mathcal{X}(\boldsymbol{\Phi})$ is the set of predicted extremal contours. Unlike other approaches ([10,7,11]), where the image contours are computed using standard methods and where the cost function is computed using the closest image edges, we compute the distance in the neighborhood of each model body part using the oriented edges obtained above. We use the chamfer distance to compute the error function. The advantage of the chamfer distance is that it does not require model-contour to image-contour assignments and its computation is fast. From the explicit parameterization of the *extremal contours*, we derive an explicit formulation of their motion and therefore we consider the distance function as a differentiable function. As a consequence, the tracking can be considered as a standard non-linear minimization process.

To summarize (see also Fig. 1): to avoid the use of silhouettes for human motion tracking, we propose a contour-based approach. The explicit (analytic) parameterization of the extremal contours of the articulated body model allows us to perform model-based edge detection, which is the first contribution of the paper. As a second contribution, we cast the tracking into a minimization problem by considering the chamfer distance as a differentiable function.

Note that a preliminary version of this work using an ad-hoc kinematic parametrization and a background substraction was described in [17]. In [17], we described a method using silhouettes and a standard edge detector (Canny-Deriche) without taking advantage of anisotropic filtering of color images. A main drawback of this work is its dependance on the silhouette estimation which often fails due to background substraction problems. In this paper we circumvent the errors introduced by flawed silhouette estimation by introducing a solely edge-based method.

Paper organization. In section 2, we recall the parameterization of extremal contours and derive their 2-D motion as given in [17]. Taking advantage of this explicit parameterization, we introduce the model-based edge detector (section 3). From the explicit parameterization and the model based detection, we derive a differentiable error function to perform the motion tracking (section 4). Finally, we discuss the method and present results in section 5.

2 The kinematics of extremal contours

We perform human motion tracking through a non linear minimization process. To perform such a minimization, one needs to compute the Jacobian of the error function or, equivalently, to estimate the motion of model points that reduces the discrepancy between the model extremal contours and the observed image contours.

Let us denote by x an image point lying onto the extremal contour of a modelled body part, let $\boldsymbol{x} = (x_1, x_2)$ be its associated coordinates and let $\boldsymbol{X} = (X_1, X_2, X_3)$ be its associated 3-D point in the body part frame. Let us also denote $\boldsymbol{X}^w$ the coordinate vector of point X in the world reference frame: $\boldsymbol{X}^w = \mathbf{R}\boldsymbol{X} + \boldsymbol{t}$, where $\mathbf{R}$ (3×3 rotation matrix) and $\boldsymbol{t}$ (translation vector) describe the motion of the body part and are parameterized by the joint parameters $\boldsymbol{\Phi}$. The motion of point x is, therefore, computed as follows:

$$\frac{d\boldsymbol{x}}{dt} = \frac{d\boldsymbol{x}}{d\boldsymbol{X}^w}\frac{d\boldsymbol{X}^w}{dt} = \mathbf{J}_I\left(\dot{\mathbf{R}}\boldsymbol{X} + \dot{\boldsymbol{t}} + \mathbf{R}\dot{\boldsymbol{X}}\right) = \mathbf{J}_I(\mathbf{A} + \mathbf{B})\begin{pmatrix}\boldsymbol{\Omega}\\ \boldsymbol{V}\end{pmatrix}, \tag{2}$$

where $(\boldsymbol{\Omega}, \boldsymbol{V})^\top = \mathbf{J}_K\dot{\boldsymbol{\Phi}}$ is the kinematic skrew. In the remaining of this section, we will make explicit each one of the terms in the equation above.

$\mathbf{J}_I$ describes the classical Jacobian of the projection transformation. We have $\boldsymbol{x} = (x_1, x_2) = (X_1^w/X_3^w, X_2^w/X_3^w)$, then

$$\mathbf{J}_I = \begin{bmatrix} 1/X_3^w & 0 & -X_1^w/(X_3^w)^2 \\ 0 & 1/X_3^w & -X_2^w/(X_3^w)^2 \end{bmatrix}. \tag{3}$$

$\mathbf{J}_K$ desribes the classical Jacobian that maps the articulated structure parameters to the body part velocities $(\boldsymbol{\Omega}, \boldsymbol{V})^\top$. One may refer to [18] for further details.

2.1 The rigid and sliding motions of extremal contours

The right-hand side of equation (2) is a transformation that allows to determine the velocity of a point from the motion of the rigid part on which this point lies. When a point is rigidly attached to the part, this transformation is given by matrix $\mathbf{A}$ (see below). In our case, as explained below, the point slides onto the smooth surface, therefore, there is a second transformation – matrix $\mathbf{B}$ – that remains to be determined.

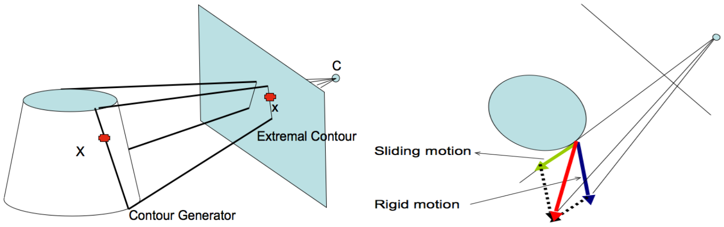
The rigid motion. The first component is computed by considering the rigid motion part of equation (2):

$$\dot{\mathbf{R}}\boldsymbol{X} + \dot{\boldsymbol{t}} = \dot{\mathbf{R}}\mathbf{R}^\top(\boldsymbol{X}^w - \boldsymbol{t}) + \dot{\boldsymbol{t}} = \mathbf{A}\begin{pmatrix}\boldsymbol{\Omega}\\ \boldsymbol{V}\end{pmatrix}, \tag{4}$$

where $\mathbf{A}$ is the $3{\times}6$ matrix that allows to compute the velocity of a point from the kinematic screw ($(\boldsymbol{\Omega}, \boldsymbol{V})^\top$) of the rigid-body motion:

$$\mathbf{A} = \left[[\boldsymbol{t} - \boldsymbol{X}^w]_\times \;\; \mathbf{I}_{3\times 3}\right]. \tag{5}$$

The notation $[\boldsymbol{m}]_\times$ stands for the skew-symmetric matrix associated with a vector $\boldsymbol{m}$.



(a) (b)

Fig. 2. (a) A truncated elliptic cone projects onto an image as a pair of *extremal contours*. The 2-D motion of these extremal contours is a function of both the motion of the cone and the sliding of the *contour generator* along the smooth surface of the cone. (b) With a perspective projection, the real contour motion differs from the rigid motion in the image.

The sliding motion. We consider the motion of an extremal contour point. Its associated 3-D point lies on a *contour generator* – the locus of points where the surface is tangent to the lines of sight (see Fig. 2-a). This tangency constraint writes:

$$(\mathbf{R}\boldsymbol{n})^\top (\mathbf{R}\boldsymbol{X} + \boldsymbol{t} - \boldsymbol{C}) = \boldsymbol{X}^T\boldsymbol{n} + (\boldsymbol{t} - C)^T\mathbf{R}\boldsymbol{n} = 0, \tag{6}$$

where vector $\boldsymbol{n}$ is normal to the surface at $\boldsymbol{X}$, and $\boldsymbol{C}$ is the camera optical center in world coordinates. $\boldsymbol{X}$ belongs to a developable surface, namely the truncated elliptic cone, parameterized by θ and z:

$$\boldsymbol{X}(\theta, z) = \begin{pmatrix} a(1+kz)\cos(\theta) \\ b(1+kz)\sin(\theta) \\ z \end{pmatrix}. \tag{7}$$

Then in equation (6), $\boldsymbol{n} = \frac{\partial \boldsymbol{X}}{\partial z} \times \frac{\partial \boldsymbol{X}}{\partial \theta} = \boldsymbol{X}_z \times \boldsymbol{X}_\theta$. For any rotation, translation, and camera position, equation (6) allows to estimate $\boldsymbol{X}$ as a function of the surface parameters. For the truncated elliptic cone i.e. equation (7), $\boldsymbol{X}$ lies on a line and therefore the extremal contours are simply a pair of lines. To compute $\dot{\boldsymbol{X}}$ we simply differentiate the tangency constraint. After some algebraic manipulations, we obtain:

$$\mathbf{R}\dot{\boldsymbol{X}} = \mathbf{B}\begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix}. \tag{8}$$

$\mathbf{B} = b^{-1}\mathbf{R}\boldsymbol{X}_\theta \ (\mathbf{R}\boldsymbol{n})^\top [[\boldsymbol{C} - \boldsymbol{t}]_\times \ \ -\mathbf{I}]$ is a 3×6 matrix with $b = (\boldsymbol{X} + \mathbf{R}^T(\boldsymbol{t} - C))^T\boldsymbol{n}_\theta$. The sliding of the contour generator is in the tangent plane and therefore tangent to the line of sight. In perspective projection, the sum of the sliding and the rigid motion projects to an image velocity which is different from the pure rigid motion (see Fig. 2-b). This sliding componnent is not taken into account in approaches based on optical flow for tracking [10].

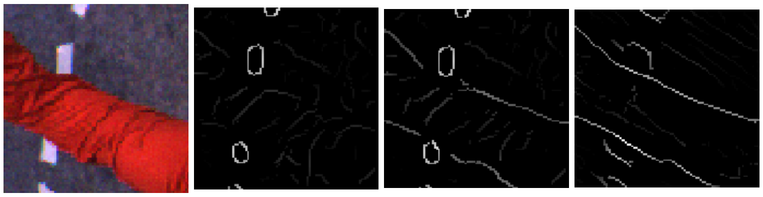


Fig. 3. (a) The arm is visible from the top view. (b) Standard Canny filter reveals markings on the ground and foldings of the clothings. (c) Color Canny filter partially detects the arm contours. (d) Anisotropic Gaussian color filter reveals the full arm contours.

3 Model-Based Contour Detection

For the human tracker to successfully track the human motion, the correct detection of the *extremal contours* in the image is essential. Both edges caused by the background, and those caused by texture within the actor, could distract the tracker from the true extremal boundaries. Therefore, the predicted model contours are used to extract edges using an edge aligned anisotropic filter in the neighborhood of the predicted extremal contour. The aligned edge detection emphasizes edges in the modelled direction while suppressing edges in undesired directions. This model-based contour detection minimizes the chance of ’false’ boundarie detection, thereby optimizing the chance of successful tracking.

Color edges. We start by detecting the color edges. Given a color image, $I\left(\boldsymbol{x}\right) = \left(R\left(\boldsymbol{x}\right), G\left(\boldsymbol{x}\right), B\left(\boldsymbol{x}\right)\right)^T$, the local differential structure is described by the color tensor,

$$\mathbf{G} = \begin{pmatrix} \overline{I_x \cdot I_x} & \overline{I_x \cdot I_y} \\ \overline{I_y \cdot I_x} & \overline{I_y \cdot I_y} \end{pmatrix}, \tag{9}$$

where I_x and I_y denote horizontal and vertical gradients, and the bar $(\bar{\cdot})$ operator denotes a convolution with a Gaussian kernel. DiZenzo [19] pointed out that the structure tensor correctly combines the vectors in the separate channels. A simple addition could lead to edge annihilation in case of opposing derivatives, whereas the principle eigenvalue of the color tensor,

$$\lambda_1 = \tfrac{1}{2}\left(\overline{I_x^2} + \overline{I_y^2} + \left(\left(\overline{I_x^2} - \overline{I_y^2}\right)^2 + \left(2\overline{I_x \cdot I_y}\right)^2\right)^{1/2}\right) \tag{10}$$

correctly detects the color edges. This prevents the disappearance of isoluminant edges as is indicated in Fig. 3.

Anisotropic filtering. To minimize the chance of undesired edges, which complicate the subsequent minimization procedure, we exploit the model information derived from the kinematic model. Based on the predictions from the kinematic model, the image is divided into patches, each of which contains a single predicted extremal contour. Then,

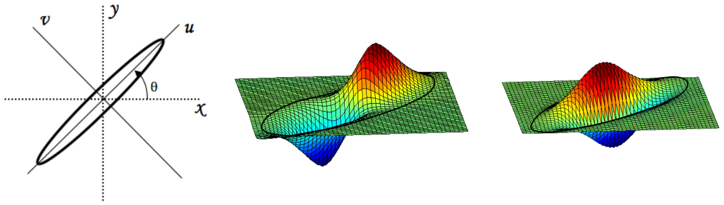


Fig. 4. (left) An example of anisotropic Gaussian with orientation $\theta = \Pi/4$; (middle) Gaussian derivative in the u direction; (right) Gaussian derivative in the v direction

from the model we derive both the length and the orientation of the predicted contour in the current image. This information is used to explicitly focus the derivative filters on edges in a particular direction. For this purpose we apply anisotropic Gaussian filtering [14], [20], for which the kernel is given by:

$$g\left(u, v; \sigma_u, \sigma_v, \psi\right) = \frac{1}{2\pi\sigma_u\sigma_v} e^{-\left(\frac{u^2}{2\sigma_u^2} + \frac{v^2}{2\sigma_v^2}\right)}, \tag{11}$$

where $(u, v)^T = \mathbf{R}(x_1, x_2)^T$ and $\mathbf{R}$ is 2×2 rotation matrix of angle ψ. The three parameters describing the anisotropic Gaussian are derived from the kinematic model. ψ is given by the orientation of the considered extremal contour. σ_u is given by the extremal contour size, $\sigma_u = length/4$ and $\sigma_v = 2$ with the constraint $\sigma_u > \sigma_v$. Once

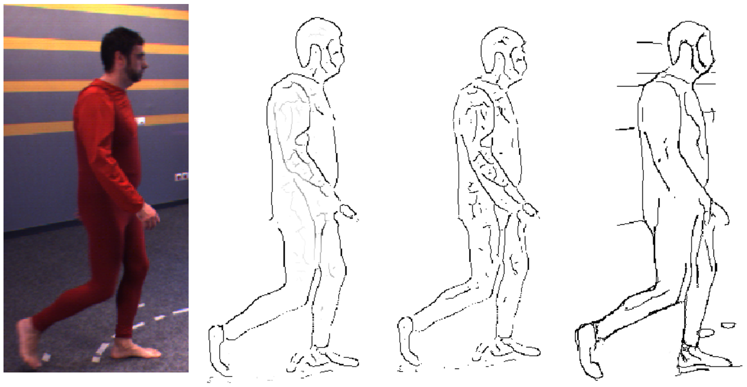


Fig. 5. (a) Original color image. (b) Standard Canny filter. (c) Color Canny filter detects the contours partially better but there still exists plenty of distracting edges. (d) Anisotropic Gaussian color filter reveals the full boundary edges.

aligned with an edge, the anisotropic filter increases smoothing along the edge, and reduces smoothing across the edge. This ensures better contrast conservation than is obtainable with isotropic filters. Moreover, responses from edges which deviate significantly from the kernel orientation are suppressed. An example of an anisotropic

Gaussian is given in Fig. 4. The anisotropic color tensor is computed by applying the derivative filters $g_u\left(u, v; \sigma_u, \sigma_v, \psi\right)$ and $g_v\left(u, v; \sigma_u, \sigma_v, \psi\right)$ to compute the derivatives I_u, I_v. The eigenvalues of the color tensor constructed with I_u and I_v describe the anisotropic color edge map of the image. Finally, we apply the color Canny algorithm as described in [16] to compute the binary edge map. Fig. 4c and d show the gain which is obtained by applying an anisotropic color Canny instead of a standard color Canny. The elongated anisotropic filter does not get distracted by the perpendicular edges of the white dashes. To efficiently compute the anisotropic Gaussian we use a recursive implementation [14], [21], [22].

4 Fitting extremal contours to image contours

In this section, we consider the problem of fitting the predicted extremal contours with image contours extracted with the method described in section 3. To perform this tracking, we have to measure the discrepancy between a set of predictions (extremal contours) and a set of observations (image contours): we want to find the model's parameters that minimize this discrepancy. For the sake of clarity of exposition we consider only one body part seen from one camera. We collect extremal-contour points from the body-part. Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j,$
$\ldots, \boldsymbol{x}_m\}$ be the prediction vector, a set of m predicted extremal-contour points. The components of this vector are 2-D points and they are parameterized by $\boldsymbol{\Phi}$. Similarly, let $\mathcal{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_i, \ldots, \boldsymbol{y}_k\}$ be the observation vector – a set of contour points observed in the image patch which contains the predicted body part extremal contour. In order to estimate the motion parameters one has to compare these two sets through a metric and to minimize it over the motion variables. Therefore, the problem can be generally stated as the minimization of a multi-variate scalar function E (equation (1)). One possible choice for the error function, that works well in practice, is the sum of the distances to the nearest image contour over all the predicted extremal contours points. This distance can be efficiently computed as a chamfer distance performed after the edge detection. Then, the error function writes:

$$E(\mathcal{Y}, \mathcal{X}(\boldsymbol{\Phi})) = \sum_{j=1}^{m} D_j^2(\mathcal{Y}, \boldsymbol{x}_j(\boldsymbol{\Phi})), \tag{12}$$

where $D_j^2(\mathcal{Y}, \boldsymbol{x}_j(\boldsymbol{\Phi}))$ is the bi-linear interpolation of the chamfer-distance image at point $\boldsymbol{x}_j(\boldsymbol{\Phi})$. We denote by $[x]$ the integer part of a real number x. Let $u_1 = [x_1]$ and $u_2 = [x_2]$ be the integer parts, and $r = x_1 - [x_1]$ and $s = x_2 - [x_2]$ be the fractional parts of the coordinates of a predicted point $\boldsymbol{x}$. $D(\mathcal{Y}, \boldsymbol{x})$ writes as:

$$\begin{aligned} D(\mathcal{Y}, \boldsymbol{x}) = \alpha C_{\mathcal{Y}}(u_1, u_2) + \beta C_{\mathcal{Y}}(u_1 + 1, u_2) \\ + \gamma C_{\mathcal{Y}}(u_1, u_2 + 1) + \lambda C_{\mathcal{Y}}(u_1 + 1, u_2 + 1), \end{aligned} \tag{13}$$

where, $\alpha = (1 - r)(1 - s)$, $\beta = r(1 - s)$, $\gamma = (1 - r)s$, $\lambda = rs$ and $C_{\mathcal{Y}}$ denotes the chamfer image computed from the extremal contour map. Note that to avoid the chamfer map to be distracted by the other edges from other body parts, we compute the chamfer map on each of the edge map patches (Fig. 2(e)).

4.1 Minimizing the chamfer distance

The minimization problem defined by equation (1) can be rewritten as the sum of squares of the chamfer distances over the predicted model contours:

$$f(\boldsymbol{\Phi}) = \frac{1}{2}\sum_{j=1}^{m} D_j^2(\mathcal{Y}, \boldsymbol{x}_j(\boldsymbol{\Phi})) = \frac{1}{2}\sum_{j=1}^{m} D_j^2(\boldsymbol{\Phi}). \tag{14}$$

In order to minimize this function over the motion parameters, we take its second-order Taylor expansion as well as the Gauss-Newton approximation of the Hessian:

$$f(\boldsymbol{\Phi} + \boldsymbol{d}) = f(\boldsymbol{\Phi}) + \boldsymbol{d}^\top \mathbf{J}_D^\top \boldsymbol{D} + \frac{1}{2}\boldsymbol{d}^\top \mathbf{J}_D^\top \mathbf{J}_D \boldsymbol{d} + \ldots,$$

where $\boldsymbol{D}^\top = \left(D_1 \, \ldots \, D_m \right)$ and $\mathbf{J}_D^\top = [d\boldsymbol{D}/d\boldsymbol{\Phi}]^\top$ is the $n \times m$ matrix:

$$\mathbf{J}_D^\top = \left[\tfrac{d\,D_1}{d\,\boldsymbol{\Phi}} \, \cdots \, \tfrac{d\,D_m}{d\,\boldsymbol{\Phi}} \right]. \tag{15}$$

The derivative of the chamfer distance D_j with respect to the motion parameters decomposes as: $\frac{d\,D_j}{d\,\boldsymbol{\Phi}} = \left[\frac{d\,D_j}{d\,\boldsymbol{x}}\right]^\top \frac{d\,\boldsymbol{x}}{d\,\boldsymbol{\Phi}}$. By noticing that $d[x]/dx = 0$, we immediately obtain an expression for $dD_j/d\boldsymbol{x}$:

$$\begin{aligned}
\frac{\partial D_j}{d\,x_1} &= (1-s)(C_{\mathcal{Y}}(u_1+1, u_2) - C_{\mathcal{Y}}(u_1, u_2)) + \\
&\quad s(C_{\mathcal{Y}}(u_1+1, u_2+1) - C_{\mathcal{Y}}(u_1, u_2+1)) \\
\frac{\partial D_j}{d\,x_2} &= (r-1)(C_{\mathcal{Y}}(u_1+1, u_2) + C_{\mathcal{Y}}(u_1, u_2)) + \\
&\quad r(C_{\mathcal{Y}}(u_1+1, u_2+1) + C_{\mathcal{Y}}(u_1, u_2+1)).
\end{aligned}$$

From equation (2), we have $\frac{d\boldsymbol{x}}{d\boldsymbol{\Phi}} = \mathbf{J}_I(\mathbf{A} + \mathbf{B})\mathbf{J}_K$. We then perform the minimization using the Levenberg-Marquardt algorithm.

5 Discussion and results

In this section we show results of our contour-based tracker. The lack of robustness we encountered with a silhouette-based approach motivated us to design a purely edge-based method. We will illustrate both the failure of standard edge detection methods and a successful tracking of a long sequence based on our model-based contour tracker.

But firstly, let's go back to the problem of minimizing the chamfer distance. At each time instant, the tracker is initialized with the previously found solution and equation (14) must be minimized. This minimization problem needs one necessary condition, namely that the $n \times n$ Hessian matrix has full rank. The Jacobian $\mathbf{J}_D$ is of size $m \times n$ and we recall that n is the number of variables to be estimated (the motion parameters) and m is the number of predictions (extremal contour points). To compute the inverse of $\mathbf{J}_D^\top \mathbf{J}_D$ we must have $m \geq n$ with n independent matrix rows.

Since each prediction accounts for one row in the Jacobian matrix, one must somehow ensure that there are n "independent" predictions. If each body part is viewed as a rigid object in motion, then it has six degrees of freedom. A set of three non-colinear points constrain these degrees of freedom. Whenever there are one-to-one model-point-to-image-point assignments, a set of three points is sufficient to constrain all six degrees of freedom. In the case of the chamfer distance there are no such one-to-one assignments and each model point yields only one constraint. Therefore, when one uses the chamfer distance, the problem is underconstrained since three non-colinear points yield three constraints only. Within a kinematic chain each body-part has p degrees of freedom. Fortunately the body-parts are linked together to form kinematic chains. Therefore, one sensible hypothesis is to assume that the points at hand are evenly distributed among the body parts.

The kinematic human-body model that we use is composed of 5 kinematic chains that share a common root body-part, 19 body-parts, and 40 degrees of freedom. Therefore, with an average of 3 points per body-part, there are in principle enough constraints to solve the tracking problem. Notice that the root-body part can arbitrarily be chosen and there is no evidence than one body-part is more suitable than another body-part to be the root part.

In practice there are other difficulties and problems. Due to total and partial occlusions the numbers of visible body-parts varies. Therefore, it is not always possible to ensure that that all the degrees of freedom are actually measured in one image. Even if a point attached to a visible body-part is predicted in the image, it may not be present in the data and/or it may be badly extracted and located. Non-relevant edges that lie in the neighborhood of a predicted location contribute to the chamfer distance and therefore complicate the task of the minimization process.

One way to increase the robustness of the tracker is to use additinal data. The latter may be obtained by using several cameras, each camera providing an independent chamfer distance error function. Provided that the cameras are *calibrated and synchronized* the method described above can be simultaneously applied to all the cameras. There will be several Jacobian matrices of the form of $\mathbf{J}_D$ (one for each camera) and these matrices can be combined together in a unique Jacobian, provided that a common world reference frame is being used [11]. Therefore, one increases the number of predictions (lines in the Jacobian) without increasing the number of variables.

It is worthwhile to notice that the extremal contours viewed with one camera are different than the extremal contours viewed with another camera. Indeed, these two sets of contours correspond to different physical points onto the surface. One great advantage of this feature is that there is no need to establish point-to-point matches between images taken with distinct cameras.

We will now illustrate the advantages of the proposed model based contour method. We performed experiments with realistic and complex human motion. The system is composed of 6 synchronized cameras running at 30 frames/second. The minimization process which resides in the inner loop of the proposed tracking approach converges in approximately 5 iterations. After 5 iterations, the optimization do not lead to worthy improvement of the estimated pose.

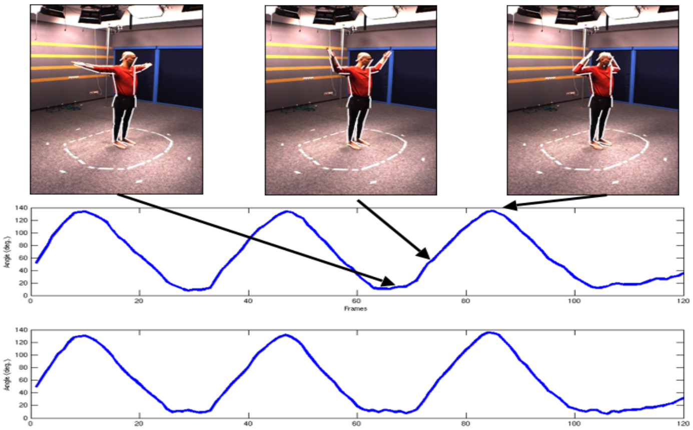


Fig. 6. Joint angles of left and right elbows during simple gymnastics.

Fig. 6 shows a plot of the angular values of both left and right elbows during the first 120 frames of a 600 frames sequence. Fig. 7 provides a comparison between the method based on standard edge detection and the method using anisotropic Gaussian filtering. Top row provides a camera view point with model contours obtained using the anisotropic gaussian filtering. Second and third rows shows the the standard edges and the model pose. The method fails when the arms are to close to the head since their are too many distracting edges. The last rows provides the model based edge detection and the model pose.For the clarity of this figure, the edges are gathered on an single image. The tracking performs well since very few distracting edges still remain.

Finally, fig. 8 provides an example of motion tracking performed on images. Top row of fig. 8 provides a camera view point. The predicted extremal contours are shown on those images. Providing this prediction, the model based edge detection is performed in the neighborhood of each extremal contour. For the clarity of this figure, the edges are gathered on an single image (middle row). Using both the prediction of the extremal contours and the edge detection we estimate the new model pose (bottom row). The tracker performed well on this 220 frames long video sequence.

6 Conclusions

In this paper we proposed a method for tracking the motion of articulated objects that combines a kinematic parameterization of the object's extremal contours with edge detection performed by an anisotropic Gaussian filter. The method relies on contour tracking, i.e., it minimizes the sum of squares of error functions between predicted model contours and image contours. This error is estimated using the *directed chamfer distance*. The advantage of the latter is that it does not need data-point-to-model-point assignments. Whenever a body part is predicted visible in an image, anisotropic edge detection is applied to an appropriate image patch and is guided by the orientation of the predicted extremal contours. This process filters out irrelevant edges, such as background edges or edges produced by clothes. The model-to-image contour fitting is carried out over all the image patches (one image patch per body parts), therefore it avoids interactions between image edges and extremal contours that should not be matched. We discussed in detail the issue of how many cameras should be used to perform articulated motion tracking and we came to the conclusion that, in principle, one camera may be sufficient. Nevertheless, an increase in the number of cameras drastically improves both the robustness of the minimizer and the quality of the results. This contour-based method compares well with a silhouette-based method simply because our contour detection method provides a richer image description. For example, it takes into account edges inside the silhouette such as the inner edge of the arm when the latter sticks to the torso. Future work will investigate ways to enforce color and motion coherence during tracking to further limit the effect of the background clutter. Currently, we invert a highly redundant set of constraints from all visible contours in all images. Another direction of research is how to select the “most attractive” image contours to further enhance our tracker.

References

1. Agarwal, A., Triggs, B.: Learning to track 3d human motion from silhouettes. In: International Conference on Machine Learning, Banff (2004) 9–16 1
2. Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. In: Computer Vision and Pattern Recognition. (2000) 2126–2133 1
3. Lan, X., Huttenlocher, D.P.: A unified spatio-temporal articulated model for tracking. In: Computer Vision and Pattern Recognition. (2004) 722–729 1
4. Bregler, C., Malik, J., Pullen, K.: Twist based acquisition and tracking of animal and human kinematics. International Journal of Computer Vision **56** (2004) 179–194 1
5. Sminchisescu, C., Triggs, B.: Estimating articulated human motion with covariance scaled sampling. International Journal of Robotics Research **22** (2003) 371–379 1
6. Delamarre, Q., Faugeras, O.: 3d articulated models and multi-view tracking with physical forces. Computer Vision and Image Understanding **81** (2001) 328–357 1
7. Gavrila, D., Davis, L.: 3d model-based tracking of humans in action: a multi-view approach. In: Conference on Computer Vision and Pattern Recognition, San Francisco CA (1996) 73–80 1, 3
8. Ilic, S., Salzmann, M., Fua, P.: Implicit surfaces make for better silhouettes. In: European Conference on Computer Vision. (2005) I: 1135–1141 1
9. Kakadiaris, I., Metaxas, D.: Model-based estimation of 3d human motion. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 1453–1459 1
10. Drummond, T., Cipolla, R.: Real-time visual tracking of complex structures. IEEE Trans. Pattern Analalysis Machine Intelligence **24** (2002) 932–946 2, 3, 5

11. Martin, F., Horaud, R.: Multiple camera tracking of rigid objects. International Journal of Robotics Research **21** (2002) 97–113 2, 3, 10
12. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Analysis and Machine Intelligence **8** (1986) 679–698 2
13. Perona, P.: Steerable-scalable kernels for edge detection and junction analysis. In: European Conference on Computer Vision. (1992) 3–18 2
14. Geusebroek, J., Smeulders, A.W.M., van de Weijer, J.: Fast anisotropic gauss filtering. IEEE Trans. Image Processing **12** (2003) 938–943 2, 7, 8
15. Ronfard, R.: Region based strategies for active contour models. International Journal of Computer Vision **13** (1994) 229–251 2
16. van de Weijer, J., Gevers, T.: Tensor based feature detection for color images. In: Proc. IS&TSID's CIC 2004, Scottsdale, Arizona, USA (2004) 2, 8
17. Knossow, D., Ronfard, R., Horaud, R., Devernay, F.: Tracking with the kinematics of extremal contours. In Narayanan, P., Shree K. Nayar, S.K., Shum, H.Y., eds.: Computer Vision – ACCV 2006. LNCS, Hyderabad, India, Springer (2006) 664–673 3
18. Murray, R., Li, Z., Sastry, S.S.: A Mathematical Introduction to Robotic Manipulation. CRC Press, Ann Arbor (1994) 4
19. Di Zenzo, S.: Note: A note on the gradient of a multi-image. Computer Vision, Graphics, and Image Processing **33** (1986) 116–125 6
20. Koenderink, J.J., van Doorn, A.J.: Receptive field families. Biol. Cybern. **63** (1990) 291–297 7
21. Triggs, B., Sdika, M.: Boundary conditions for young - van vliet recursive filtering. To appear in IEEE Transactions on Signal Processing (2006) 8
22. Young, I.T., van Vliet, L.J.: Recursive implementation of the gaussian filter, signal processing. Signal Processing **44** (1995) 139–151 8

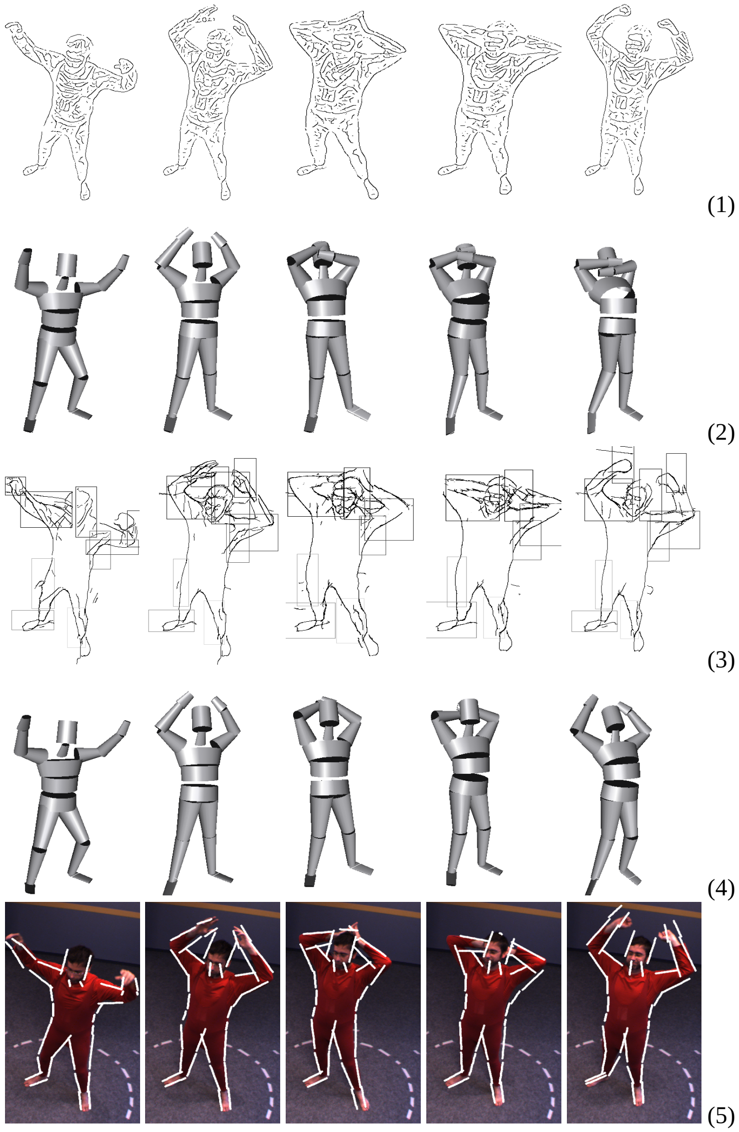


Fig. 7. Tracking of a 200-frame video sequence comparing standard edge detection method with the Anisotrpic Gaussian filtering. The standard edge detection on gray level images is shown on row (1). The estimated pose is given on row (2). Note that the tracker fails: the arms are not correctly tracked. The Anisotrpic Gaussian filtering (3) performs well on the arms compared to the standard method. The estimated pose is given on row (4). Last row (5) shows the extremal contours projected onto the original camera images.

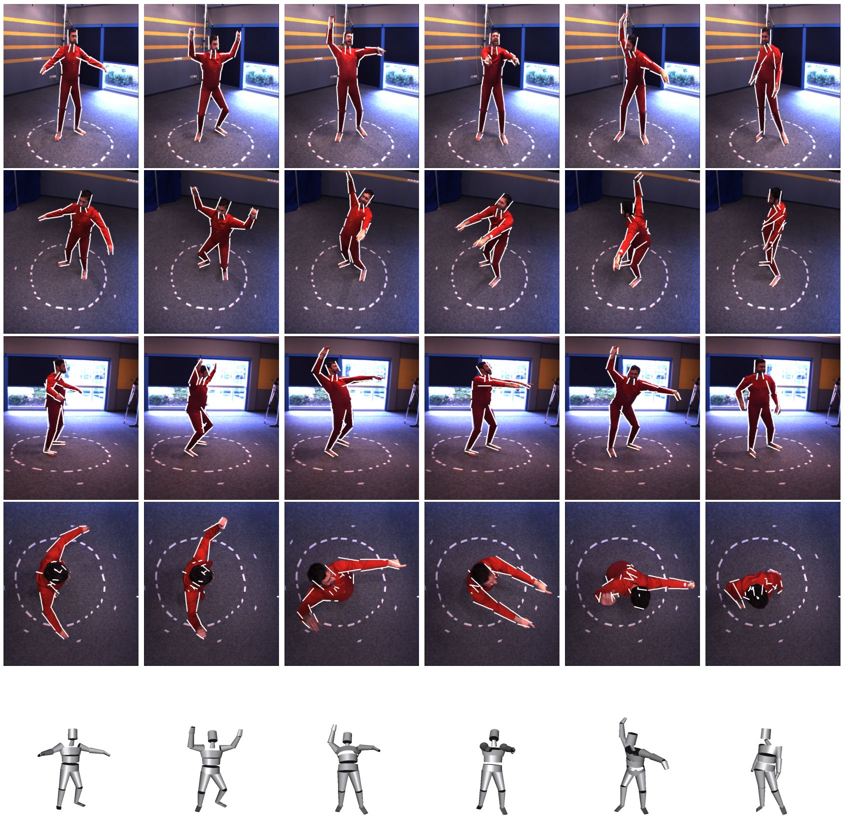


Fig. 8. Tracking of 250 frames video sequence. Each first four rows shows one camera viewpoint. The extremal contours are predicted and shown on each camera image. Using this prediction, we perform the model-based tracking to obtain the new model pose (bottom row).