Bootstrapping the Condensation Algorithm

Thomas B. Moeslund and Erik Granum
Laboratory of Computer Vision and Media Technology
Aalborg University, Denmark
{tbm,eg}@cvmt.dk

Abstract

In model-based tracking the problem of a high dimensional solution space often appears. The standard solution to this problem is to use a prediction followed by an iterative search or a Kalman filter. The drawback of both is the risk of ending up in a local extremum. In this paper we suggest to apply bootstrapping to increase performance in model-based tracking. The bootstrapping information is in the form of the position of the hand in the image. The idea of bootstrapped tracking is exemplified in the context of monocular tracking of the 3D pose of a human arm utilising the Condensation algorithm. A number of tests are conducted and it is concluded that bootstrapped tracking is a promising approach when solving some of the inherent problems in model-based tracking.

1 Introduction

In model-based tracking the problem of a high dimensional solution space often appears. A brute force search is seldom possible due to a high number of possible solutions and hence heavy computational demands. The standard solution to this problem is to use a prediction to narrow down the solution. Either in the context of a state update, e.g. via the Kalman filter, or via an iterative algorithm. The former is when the current state of the tracked object is found by the weighted sum of the estimated state found from image data and the weighted predicted state. The latter starts with a predicted state and iteratively compares neighbour states from the solution space with the image measurement until a local extrema is reached.

Both approached are likely - sooner or later - to get stuck in a local extremum in the solution space, due to clutter in the background, wrong prediction, or when no image measurements of the tracked object can be found. Furthermore, when a monocular setup is applied to track a 3D object the image measurements become ambiguous as only one state of the object is predicted.

To handle the above mentioned problems different multi hypothesis trackers have been suggested. Most well-known is perhaps the Condensation algorithm [5]. The Condensation algorithm approximates a brute force search by predicting only the most likely states of the object in the previous image. These predictions are then compare with the image measurement to estimate the a posterior probability for the different states given the current image measurements. How good this principle works depends on two things: i) the number of samples (denoted particles) used to estimate the posterior probability, and ii) the quality of the predictions. The former can be tuned to a particular application or even changed during processing. The latter, however, is a far more tricky business!

1.1 The Content of this Paper

In this paper we will describe how bootstrapping can be used to improve the prediction in a tracking algorithm. In section 2 we will in general terms describe how bootstrapping can be applied in this context. The rest of the paper will be devoted to exemplify our idea in the context of tracking the 3D pose of a human arm by one camera utilising the Condensation algorithm. Concretely section 3 describes the geometric model of the arm utilised in this work. In section 4 we describe the representation of the image data that will be used in a comparison with the predicted states of the solution space. In section 5 we describe how the bootstrapping is utilised in our context. Section 6 presents the results and section 7 discusses our findings.

2 Prediction and Bootstrapping

Predicting a state, $\overrightarrow{U}(t)$, from time, $t-1$, to time t can be done in many ways. However, it is usually done by adding a deterministic part, $\overrightarrow{D}(T)$, and a stochastic part, $\overrightarrow{S}(T)$, hence $\overrightarrow{U}(t) = \overrightarrow{D}(T) + \overrightarrow{S}(T)$, where T indicates dependencies on all the past and t indicates a particular instance of time. $\overrightarrow{D}(T)$ consists of a motion model,

$\overrightarrow{M}(T)$, which describes how the state evolves over time. $\overrightarrow{M}(T)$ contains a number of parameters those current values are kept in $\overrightarrow{\omega}(T)$. $\overrightarrow{M}(T)$ is usually independent on time. $\overrightarrow{\omega}(T)$ is typically estimated in a recursive framework where it is assumed to be a first order Markov process, hence $\overrightarrow{\omega}(T) = \overrightarrow{\omega}(t-1)$. In praxis the deterministic part is normally defined as $\overrightarrow{D}(T) = \overrightarrow{D}(\overrightarrow{M}, \overrightarrow{\omega}(t-1))$ [2].

Very seldom can a motion model be set up that completely describes all aspects of how the state evolves over time. The stochastic part is therefore added. It models the errors in the motion model and are referred to as the process noise. $\overrightarrow{S}(T) = \overrightarrow{S}(\overrightarrow{N}(T), \overrightarrow{\phi}(T))$ where $\overrightarrow{N}(T)$ is the model of the process noise and $\overrightarrow{\phi}(T)$ is the current values of the parameters in this model. The process noise is often assumed to be independent of time and modelled as a Gaussian distribution. And as above a first order Markov process is assumed, hence $\overrightarrow{\phi}(T) = \overrightarrow{\phi}(t-1)$. In praxis the stochastic part is therefore normally defined as $\overrightarrow{S}(T) = \overrightarrow{S}(\overrightarrow{N}, \overrightarrow{\phi}(t-1))$, where $\overrightarrow{N}$ is a multivariate Gaussian distribution.

The motion model and the different parameters can be learned through training, see e.g. [2]. Learning the parameters through training can be difficult due to lack of groundtruth data. An alternative is therefore to estimate $\overrightarrow{\omega}(t-1)$ and $\overrightarrow{\phi}(t-1)$ for each image. However, this can also be very problematic. They require complete knowledge of the state at $t-1$ and as multiple hypothesise do occur finding *the* correct state for each image is not always easy. The standard way of finding the correct state is by estimating the maximum a posteriori (MAP) and this is not easy in the context of the Condensation algorithm as we shall see later.

2.1 Bootstrapping

When tracking an object it is sometimes possible to recognise parts of the object prior to tracking. For example, in the context of tracking the 3D human figure in a monocular image sequence it is in general difficult to find robust features to track, however, some features can actually be tracked independent of others. These are: the face/head, the hands, the feet, and in some cases also other distinct points, e.g. arm pits, shoulders, and crotch.

Say we are able to find one of these features, denoted $\overrightarrow{\beta}(t)$. This would allow a comparison between $\overrightarrow{\beta}(t)$ and $\overrightarrow{D}(t)$ estimating (parts of) the prediction error, hence $\overrightarrow{\phi}(t)$. Applying $\overrightarrow{\phi}(t)$ as oppose to $\overrightarrow{\phi}(t-1)$ obviously gives a far better estimate of the stochastic part. We denote the new estimate with a plus, $\overrightarrow{S}(T)^{+} = \overrightarrow{S}(\overrightarrow{N}, \overrightarrow{\phi}(t))$.

Furthermore, $\overrightarrow{\beta}(t)$ also contains information that can be used to bias the deterministic prediction, or more precisely $\overrightarrow{\beta}(t)$ can correct (parts of) the predicted state. That is, given $\overrightarrow{\beta}(t)$ we can estimate $\overrightarrow{\omega}(t)$ and apply this instead of $\overrightarrow{\omega}(t-1)$. We denote the new estimate of the deterministic part as $\overrightarrow{D}(T)^{+} = \overrightarrow{D}(\overrightarrow{M}, \overrightarrow{\omega}(t))$.

So instead of predictions based on estimates at time $t-1$ we now use our estimates from time t, $\overrightarrow{\beta}(t)$, to correct our predictions. Altogether providing a far better result.

We denote this approach *bootstrapped tracking*. The success of this approach depends on how much information is carried in $\overrightarrow{\beta}(t)$, hence how many of the state's parameters can be corrected, and how much this information can prune the solution space.

As bootstrapped tracking delivers good predictions (corrections) simple motion models can be applied. The often difficult and tiresome task of acquiring training data and learning the motion model and its parameters can thus be avoided altogether.

3 Modelling the Arm

In this work we estimate the position of the hand in the images, $[h_x, h_y]^T$, using colour segmentation [9] and let this be our bootstrapping information, hence $\overrightarrow{\beta}(t) = [h_x, h_y]^T$. In the context of $\overrightarrow{\beta}(t)$ a geometric model of the arm needs to be defined. Before doing so we introduce the assumption that the hand is a part of the lower arm and that the 3D position of the shoulder is known, e.g. via the algorithm described in [9].

The human arm is usually modelled as either the 3D positions of the elbow and hand, or by four angles together with the length of the upper arm (A_u) and the lower arm (A_l). Both representations require six parameters. If we however assume the length of the two arm segments to be known, we only need four angular parameters. These can be e.g. globographic angles, angles around fixed axes, or Euler angles [8]. The latter is very popular as it is similar to the anatomic joint angles of the arm. Ignoring the kinematic constraints of the arm the Euler representation produces a solution space with around $1.68 \cdot 10^{10}$ different solutions given an angle resolution of one degree in each dimension [8].

In our context of bootstrapped tracking we wish to derive a much more compact representation to limit the size of the solution space that is to be estimated by the Condensation algorithm. We obtain this through our bootstrapping information $\overrightarrow{\beta}(t)$ together with a different representation of the arm - the screw axis representation.

The screw axis representation is not directly related to the anatomic joints. Nevertheless it has the same ability to represent the different arm configurations as the Euler angles have. The representation are applied in robotics and computer graphics and recently also in the computer vision community [3] [4].

The representation is based on Chasles' theorem [13] which loosely states that a transformation between two coordinate systems can be expressed as a rotation around an axis, called the screw axis (or helical axis), and a translation parallel to the screw axis. In the context of modelling the human arm the screw axis is defined as the vector spanned by the shoulder and the hand. The position of the elbow is defined as a rotation, α, of an initial elbow position around the screw axis. As the length of the upper and lower arm are fixed no translation is required parallel to the screw axis, and the perpendicular distance from the elbow to the screw axis is independent on α and can be calculated without adding additional parameters. Altogether the representation requires four parameters. Three for the position of the hand, H_x, H_y, and H_z, to define the screw axis and one for the rotation around the screw axis, α. The latter is sometimes referred to as the swivel angle [11, 12]. The parameters are illustrated in figure 1.

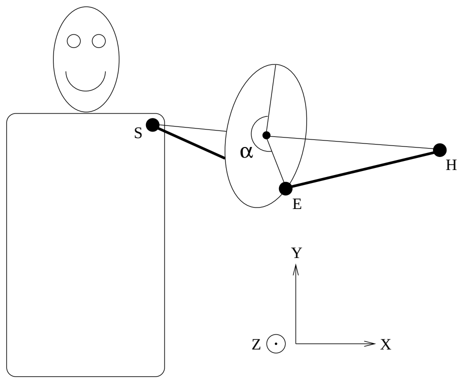

Figure 1. The screw axis representation of the human arm. α is the angle between the top point on the circle (see text for definition) and the actual elbow position. S, E, and H represent the 3D shoulder, elbow, and hand positions, respectively.

3.1 Bootstrapping the Screw Axis Representation

Little is gained so far in terms of achieving a more compact solution space. However, by enhancing the screw axis representation by the bootstrapping information, $\overrightarrow{\beta}(t)$, the size of the solution space can be reduced significantly. Combining $\overrightarrow{\beta}(t)$ with the camera parameters, $\overrightarrow{\Phi}$, obtained during calibration, the position of the hand in the image can be mapped to a line, l, in space passing through the hand, see figure 2.A. That is, the centroid of the hand is found in each image and mapped to a 3D line as

$$\overrightarrow{H}(t, \overrightarrow{\beta}(t), \overrightarrow{\Phi}) = \overrightarrow{P}(\overrightarrow{\Phi}) + t \cdot \overrightarrow{F}(\overrightarrow{\beta}(t), \overrightarrow{\Phi}) \qquad (1)$$

where $\overrightarrow{P}(\overrightarrow{\Phi})$ is the optical centre of the camera, and $\overrightarrow{F}(\overrightarrow{\beta}(t), \overrightarrow{\Phi})$ is the unit direction vector of the line. Equation 1 only contains one free parameter, namely t. In this work, however, H_z is used as the free parameters since it has a more intuitive interpretation. So for each value of H_z t and therefore also H_x and H_y are uniquely determined. By applying equation 1 to the screw axis representation we can eliminate the parameters H_x and H_y which leave us with just two parameters, namely α and H_z to model the configuration of the arm. We denote this novel model the *local screw axis model*. For each new image a unique instance of the solution space exist, hence the name "local", as opposed to the global solution spaces utilised in all of the above representations. In figure 2.B the solution space is illustrated in 3D for one image where the hand is located somewhere on the line, l. In figure 2.C (ignoring the dotted lines) the local solution space is illustarted for the parameters of the local screw axis model. Clearly this model has a very simple and compact representation of the solution space compared to the complex shape of the solution space in figure 2.B.

In this compact representation α is bounded by one circle-sweep ($0° - 360°$) while H_z is bounded by $\pm$ the total length of the arm. Given a total arm length of $60cm$ and a resolution of $1°$ for α and $1cm$ for H_z we have a solution space containing $4.32 \cdot 10^4$ different solutions. A large reduction in the size compared to that produced by Euler angles, but still a large space. We therefore introduce kinematic constraints to prune the solution space. For example, the arm can not bend backwards at the elbow. These constraints result in a minimum pruning effect of 75% and an average pruning effect of 91.8% [7] - illustrated in figure 2.C as the region within the dotted lines.

4 Image- and Object Representations

As the context of this work is model-based tracking we need a way of comparing the image data and the model data. A large part of this problem is to find a suitable representation of the both the model and image data. The two representations need to be similar in order to do the comparison. Typical image representations are 2D anatomic points (hands, head, feet, elbow, etc.), edges, silhouettes and 2D skeleton. Typical object representations are 3D anatomic points, 3D skeletons, 2D patches, and volumetric body parts.

In general it can be stated that the higher level the representation of the image is the simpler the object representation needs to be. For example, in the work by [1] the images are processed until the skeleton of the human figure is extracted, hence a high level representation allowing a simple

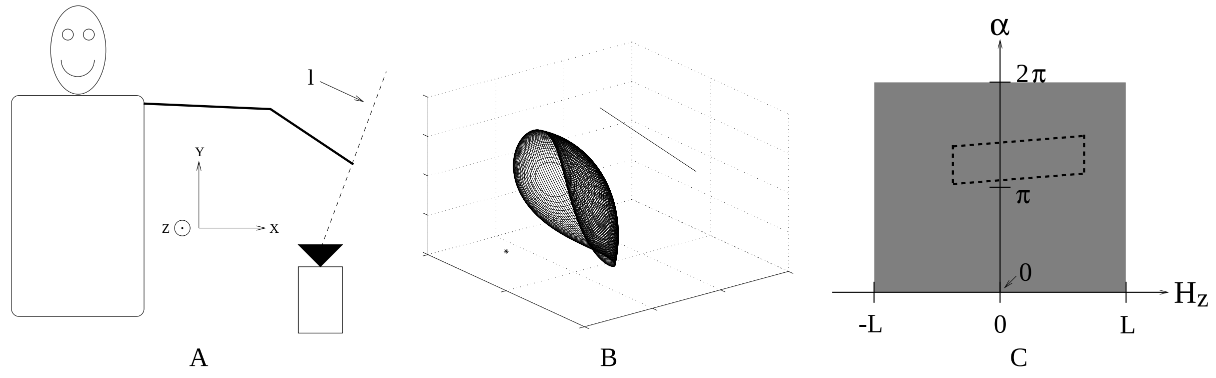

Figure 2. A: The concept utilised in the local screw axis model. B: The solution space for one image in a 3D space. The line indicates the possible positions of the hand. The "surface" illustrates the solution space for 80 discrete position of the hand on the line, hence 80 circles. The '*' is the position of the shoulder. C: The solution space in the local screw axis model, where $L = A_u + A_l$. The region within the dotted lines illustrates a typical solution space after pruning.

skeleton representation of the object model. In the work by [6] the silhouette of the human figure is used as low level image representation. To compare object model data with the silhouette a high level volumetric model of the object is required.

In this work we use a high level image representation and a low level object representation. We represent the image data by the orientations of the upper and lower arm in the image, hence a high level representation. The local screw axis model need not be enhanced as it can be mapped directly into orientations in the image given the calibration parameters, hence a low level representation.

4.1 Estimating the Orientations in the Image

We estimate the orientations of the upper arm, θ_u, and lower arm, θ_l, respectively, based on edge pixels. As our input images contain background clutter and non-trivial clothes we utilise temporal edge pixels. Hence, we find the edge pixels in the current image using a standard edge detector and AND this result with the difference image achieved by comparing the current- and the previous image.

As we wish to estimate θ_u and θ_l independently we separate the temporal edge pixels into two groups, one for the upper arm and one for the lower arm. This is done by calculating the perpendicular distance from each pixel to two hypothetical lines, one of the upper arm and one for the lower arm. The lines are defined from the position of the shoulder and hand in the image, together with a predicted position of the elbow. As the prediction of the position of the elbow is uncertain we ignore all pixels within a certain Mahalanobis distance from the predicted position of the elbow. Furthermore we ignore all pixels too far away from both lines. When no predictions are available different possible position of the predicted elbow are investigated until two representative groups are obtained. In figure 3.A a typical input image is shown. In figure 3.B the temporal edge pixels are shown.

Estimating the parameters of a straight line from data can be carried out in different ways, e.g. via PCA or linear regression. However, as we will not model the distribution via a Gaussian we can not applied these methods. Instead we apply a dynamic variant of the Hough Transform - the Dynamic Hough Transform (DHT). It estimates the likelihood of each possible orientation, hence allowing multiple peaks. The choice of the DHT is furthermore motivated by the fact that it adapts to the data. The DHT randomly samples two pixels from one group and calculates the orientation of the line spanned by the two pixels. The more times the groups are sampled the better the estimation of the distributions. On the other hand large sampling also leads to large processing time as is the case for the standard Hough Transform. The sampling is therefore terminated as soon as the variance of the distribution is stable. To evaluate the stability of the variance after n samples the variance of the last j variances is calculated as

$$\nu_{jn}^2 = \frac{1}{j} \sum_{i=n-j}^{n} (\sigma_i^2 - \mu_{jn})^2 \qquad (2)$$

where σ_i^2 is the variance after i samples and μ_{jn} is the mean of the last j variances.

The stop criterion is defined as the number of samples, n, where the last j samples are within the interval $[\mu_{jn} - \lambda, \mu_{jn} + \lambda]$. The distribution of the last j variances

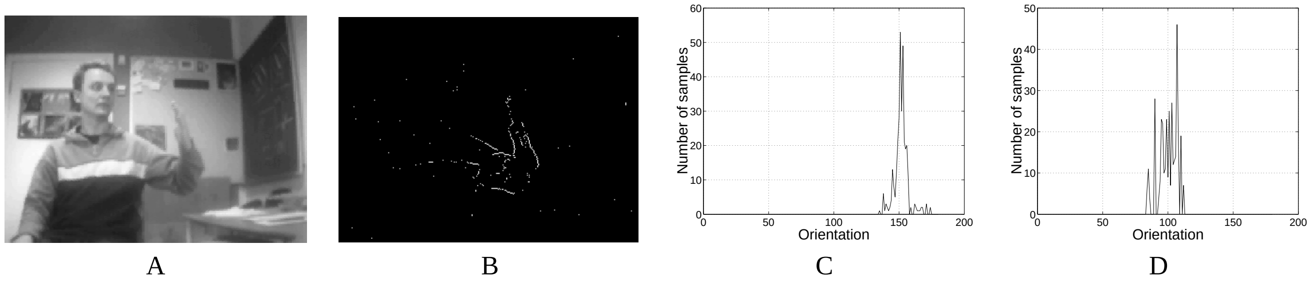

Figure 3. A: A typical input image (shown in B/W). B: The temporal edge pixels. C: The distribution of the orientation of the upper arm. D: The distribution of the orientation of the lower arm.

will in general follow a Uniform distribution. The theoretical variance of such a distribution in the given interval can be estimated as $\lambda^2/12$ [10]. When the mean of the variances, μ_{jn} is large it indicates large uncertainty in the distribution, which again indicates weak lines in the temporal edge image. A stable variance for such a distribution tends to required a larger value of λ compared to an image with stronger lines. To account for this difference λ is defined with respect to μ_{jn} as

$$\lambda = \frac{\mu_{jn}}{\gamma} \tag{3}$$

where γ is found empirically. Setting the estimated variance equal to the theoretical variance yields $\lambda = \nu_{jn}\sqrt{12}$. Inserting this result into equation 3 and write it as an inequality yields

$$\nu_{jn}^2 \leq \frac{\mu_{jn}^2}{12 \cdot \gamma^2} \tag{4}$$

Altogether the stop criterion is found as the smallest n for which inequality 4 is true. To speed up the calculations the variance is not recalculated after each new sampling, but rather for every 10th sampling.

Using the above described procedure we obtain two independent distributions, one for the upper arm, $P_u(\theta_u)$, and one for the lower arm, $P_l(\theta_l)$. Examples of these are illustrated in the figures 3.C and 3.D. Different number of samples might have been used to estimate the two distributions. The accumulated probability mass for each distribution is therefore normalised to 1. In terms of the Condensation algorithm the two normalised probability distributions are the weighting functions, hence the observation distribution, see below.

5 Bootstrapping the Condensation Algorithm

The Condensation algorithm is defined in terms of Bayes rule. That is, the posterior is equal to the observation distribution multiplied by the prior, where the prior is the predicted posterior from time $t-1$:

$$p_t(\overrightarrow{\theta}|I) = p_t(I|\overrightarrow{\theta}) \cdot p_{t-1}(\overrightarrow{\theta}) \tag{5}$$

where $\overrightarrow{\theta} = [\theta_u, \theta_l]^T$ and I is the current image. As the solution space is often too large (even when applying the local screw axis model) to conduct a brute force search the observation distribution is sampled according to the "sampling of importants" principle, hence the most likely predictions are sampled. So, as explained earlier, the key issue is to have a good prediction. Referring back to section 3 we have $\overrightarrow{\beta}(t)$ equal to the position of the hand in the image. To apply bootstrapping we need to define $\overrightarrow{D}(T)^+$ and $\overrightarrow{S}(T)^+$. We will do this first for the position of the hand, $\overrightarrow{H}$, and then for the position of the elbow, $\overrightarrow{E}$.

The correction of the prediction of $\overrightarrow{H}$ is based on the idea of combining the predictions and the image measurements. In figure 4 the predictions are illustrated using subscript 'p' while the corrected predictions are illustrated using subscript 'c'.

Since we know the camera ray through the hand in the current image, l, we can correct the prediction by projecting the predicted position of the hand, $\overrightarrow{H_p}$, to the line, l. The corrected prediction is denoted $\overrightarrow{H_c}$ and calculated as $\overrightarrow{H_c} = \overrightarrow{P} + ((\overrightarrow{H_p} - \overrightarrow{P}) \cdot \overrightarrow{F})\overrightarrow{F}$ where $\overrightarrow{P}$ and $\overrightarrow{F}$ are the line parameters defined in equation 1. The difference between the predicted and corrected vectors yields a measure of the prediction error, denoted $\overrightarrow{H_e}$ and calculated as $\overrightarrow{H_e} = \overrightarrow{H_c} - \overrightarrow{H_p}$.

The stochastic prediction models the process noise and allows a diffusion of the deterministic prediction. As we know the hand is on the line, l, we diffuse it by randomly sampling from a Gaussian distribution located along the line, l, with mean at $\overrightarrow{H_c}$ and the standard deviation controlled by the error vector, hence standard deviation $= k_1 \cdot ||\overrightarrow{H_p} - \overrightarrow{H_c}||$ where k_1 is a predefined constant.

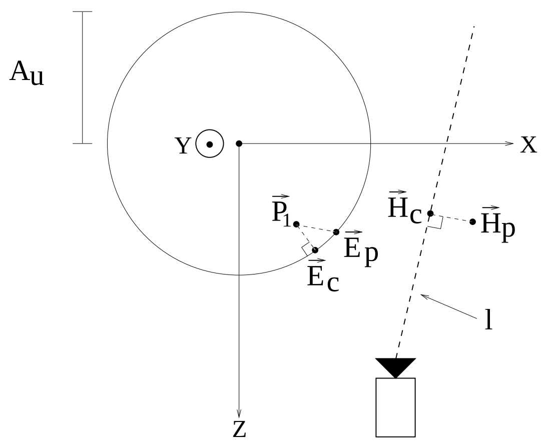

Figure 4. The shoulder coordinate system seen from above. The circle illustrates the limit of the elbow position. The dotted line indicates a camera ray through the hand. See text for a definition of the parameters.

The predicted position of the elbow can not directly be bootstrapped by $\overrightarrow{\beta}(t)$. However, we know the elbow is bound to be on a sphere with centre in the shoulder and radius equal to A_u, see figure 4. Furthermore it is likely to have an error vector closely related to that of the hand as the hand and elbow are part of the same open-looped kinematic chain. We therefore calculate the corrected position, $\overrightarrow{E_c}$, by first adding the error vector of the hand to the predicted value of the elbow, yielding $\overrightarrow{P_1} = \overrightarrow{E_p} + \overrightarrow{H_e}$. Then we project $\overrightarrow{P_1}$ to the surface of the sphere, as indicated in figure 4. The corrected elbow position is now calculated as $\overrightarrow{E_c} = \frac{\overrightarrow{P_1}}{||\overrightarrow{P_1}||}A_u$.

As the error vector has already been subjected to diffusion we do not introduce yet another diffusion of the position of the elbow.

Evidently the prior in the Condensation algorithm will be much more accurate when applying the bootstrapping compared to the standard approach. And this is true even with a very simple motion model. This observation results in two things. Firstly, we use a first order linear motion model for both the hand and the elbow, hence $x(t) = x(t-1)+d(t-1)$, where the latter term is the displacement in the previous image. This circumvent the often complicated task of acquiring sufficient ground truth training data and learning the model. Secondly, we conduct prediction at particle level, i.e. different motion parameters (displacements) for each particle. Obviously this means that the parameters of the motion model do not need to be learned, hence no training. Also, this procedure allows better predictions as the parameters of the motion model are local, hence avoiding the relationship between the estimated MAP and the prediction. This is good because the MAP is, in general, hard to estimate (more details below).

5.1 Summary of Algorithm

Too clarify our approach the different steps in the algorithm are described below. Note that the posterior is represented as an unordered list of particles, each containing the following information: $\overrightarrow{a_i} = [\pi_i, c_i, \alpha_i, \overrightarrow{H_i}, \overrightarrow{V_i}, \overrightarrow{E_i}]$, where π is the weight, c is the accumulated weight, α is the first parameter in the screw axis model, $\overrightarrow{H}$ is the 3D position of the hand, $\overrightarrow{V}$ is the displacement of the hand, and $\overrightarrow{E}$ is the 3D position of the elbow. For each particle the following is done:

1. **Sampling**
   One particle is sampled from the posterior at time $t-1$. The sampling is carried out according to the probability of each sample, hence its weight, π_i, and accumulated weight, c_i, see [5] for further details.

2. **Prediction**
   The position of $\overrightarrow{E_i}$ and $\overrightarrow{H_i}$ are predicted using a motion model with local parameters, hence $\overrightarrow{V_i}$.

3. **Bootstrapping**
   The predicted values of the hand and elbow are corrected according to $\overrightarrow{\beta}(t)$ as explained above. The predicted values are mapped into the local screw axis model (α, H_z) and checked against the pruned part of the solution space, see figure 2.C. The local parameters of the motion model are calculated, hence $\overrightarrow{V_i}$ is updated.

4. **Weighting**
   The corrected positions of the hand and elbow are mapped to two orientations in the image, one for the upper arm, θ_u, and one for the lower arm, θ_l. The weight of this particular particle is now calculated as $\pi_i = P_u(\theta_u) \cdot P_l(\theta_l)$.

After the above steps have been conducted N times the weights are normalised, $\sum \pi_i = 1$, and the cumulative probabilities, c_i, are calculated as $c_0 = 0$, $c_i = c_{i-1} + \pi_i$.

Step three is the way the bootstrapping approach enhances the original Condensation algorithm.

5.2 Issues in the Condensation Algorithm

Three issues always appear when implementing the Condensation algorithm. Firstly, the value of N needs to be decided, secondly a procedure for initialising the Condensation algorithm is required, and thirdly a procedure for estimating the MAP is required.

As already mentioned N is application dependent and can be found through test. By initialising the Condensation algorithm we mean to define the prior in the first image where no predictions are available. We delay the Condensation algorithm one image and find the posterior in the first image using our bootstrapping information. We use $\beta(0)$ to prune (α, H_z) and weight each non-pruned parameter set equally. This procedure forces the algorithm to converge much faster than given equal weights to all possible instances in the solution space and especially if the solution space is defined by the Euler angles or any other four-parameter representation.

The standard way of estimating the MAP is by a weighted average of (α, H_z) for all sampled particles. This is simple - yet un-useful - as the result might easily be located in a region where no samples are present or, even worse, in a pruned region of the solution space. Furthermore, this approach goes against the entire idea of allowing multiple hypothesises, as a weighted average is only useful for unimodal distributions.

Instead we suggest to define the actual state of the arm (MAP) as the particle the state of which minimises

$$\min_{\overrightarrow{a_i}} \sum_{j=1}^{N} \frac{1}{\pi_i} \left[\sqrt{(\alpha_i - \alpha_j)^2 + (H_{zi} - H_{zj})^2} \right] \qquad (6)$$

This expression finds the optimal solution in terms of the weighted sum of squared differences. A far better estimate of the MAP compared to the classical MAP based on the weight average. However, expression 6 is rather computational demanding. As N increases the number of computations in 6 increase with N^2. To avoid a computational explosion only those particles having a large weight, suggesting that they are part of a large peak, are investigated. That is $\pi_i > \frac{k}{N}$. The constant k directly determines the number of particles to be investigated, hence the computational complexity. Only processing the particles having a large weight do not guaranty the optimal solution. In general, however, it will be a good approximation of expression 6 and it also has the desirable feature of expression 6, namely that it insures that the estimated MAP always is in a non-pruned location.

6 Results

In this section we will show some results related to the different topics described in this paper. These are the dynamic stop criterion for the DHT and the procedure for estimating the MAP. In the last section we will discuss the results.

In figure 5 the distributions of the orientations of the upper- and lower arm are shown for the same image but for different number of samples, n. In figure 6 and 7 the variances of the distributions for $n \in [0, 10000]$ are shown.

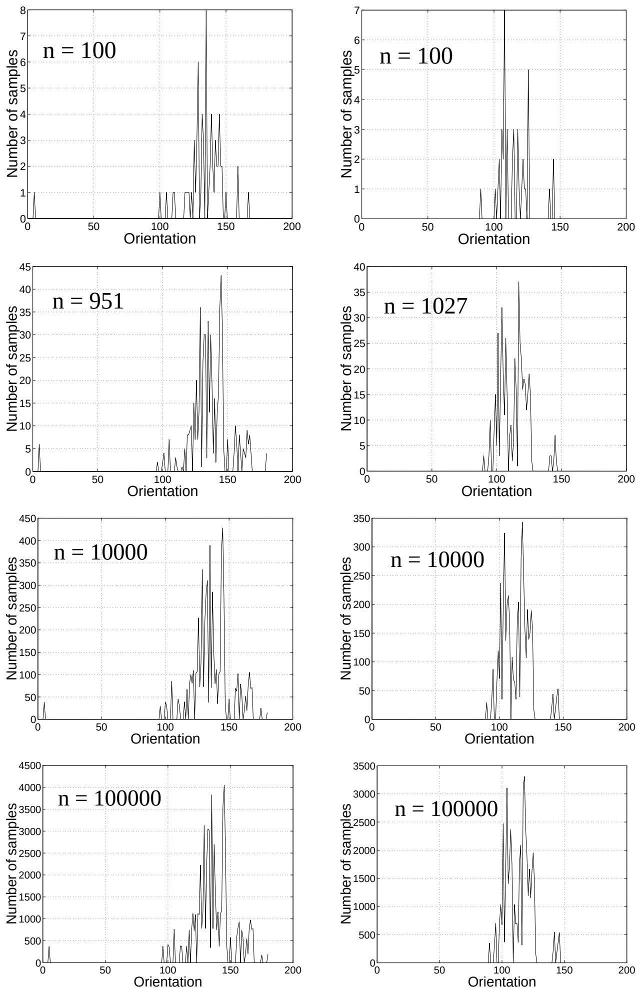

Figure 5. The estimated distributions after n samples for a particular image. The left column is for the upper arm and the right column is for the lower arm.

The final test is conducted to investigate the proposed way of estimating the MAP. In figure 8 the posterior for the image in figure 3.A is shown. The MAP estimated in the standard way using a weighted average is illustrated by a star. The approach for estimating the MAP suggested in this paper is illustrated with a diamond. Both are elevated to the same altitude of the maximum peak to improve visibility.

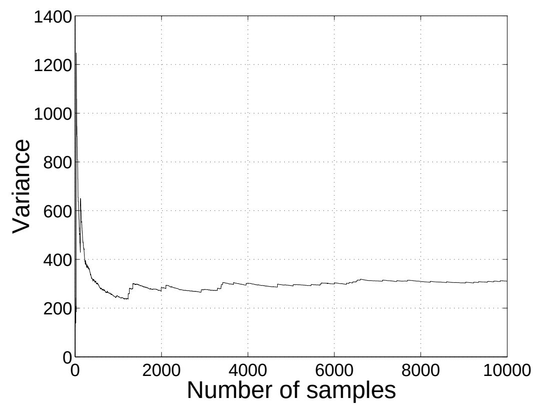

Figure 6. The variance for the distribution of the upper arm as a function of the number of samples.

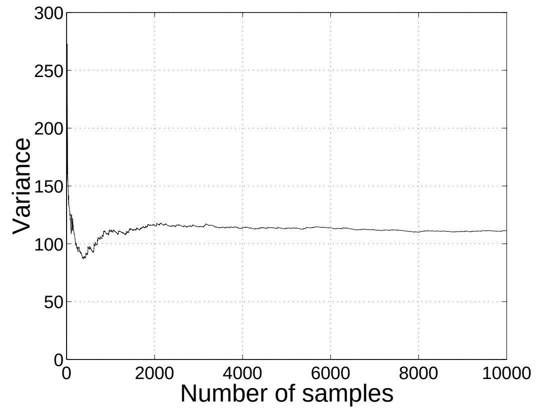

Figure 7. The variance for the distribution of the lower arm as a function of the number of samples.

7 Discussion and Conclusion

In this paper we have suggested to use distinct image features to bootstrap model-based tracking. Our idea is tested in the context of monocular tracking of the 3D pose of a human arm using the Condensation algorithm.

First we showed in general terms how to bootstrap the deterministic and stochastic parts utilised in prediction. Then we showed how the position of the hand in the image can be used to bootstrap the model representation of the arm, yielding the local screw axis model. Later we showed how the predicted parameters of the model can be corrected according to the bootstrapping information and hereby providing a better prediction. The above was implemented in the Condensation algorithm where the image measurements were in the form of orientations of the upper- and lower arm, respectively. These were estimated using the dynamic Hough Transform (DHT) with a dynamic stop criterion based on the variance of the variances. Finally we suggested a new way of estimating the MAP which takes the multi modal distribution of the posterior into account.

To justify our ideas some tests were conducted. First, the dynamic stop criterion used in the DHT was tested. In figure 5, 6, and 7 the stop criterion was tested. The tests clearly show that the variances converge suggesting the variance of the variances to be a good indicator to build the stop criterion upon. The distributions in the second row in figure 5 show the estimated distributions according to the stop criterion. Assuming the distributions in the last row to be the ”ground truth” a visual comparison suggests that the distributions estimated by the stop criterion are not identical to the ground truth. Nevertheless, it is evident that the primary tendencies are kept even though only a fraction of the samples have been applied, hence the dynamic stop criterion is valid. The choice of j and γ directly controls the computational complexity of the DHT. In this work we have utilised $\gamma = 0.05$ and $j = 100$, resulting in $n \in [500, 1500]$. Obviously these values need to be tuned to the application.

The value of N is found through tests related to the particular application. An improvement might be obtained if the value of N is changed in each image according to the current need. One way of controlling N could be to let it depend on the uncertainty in the DHT, hence n. Whether this is a solid approach will be investigated in future work.

The final test concerns the estimated MAP. The standard estimate of the MAP using a weighted average is illustrated by a star and the new method is illustrated with a diamond.

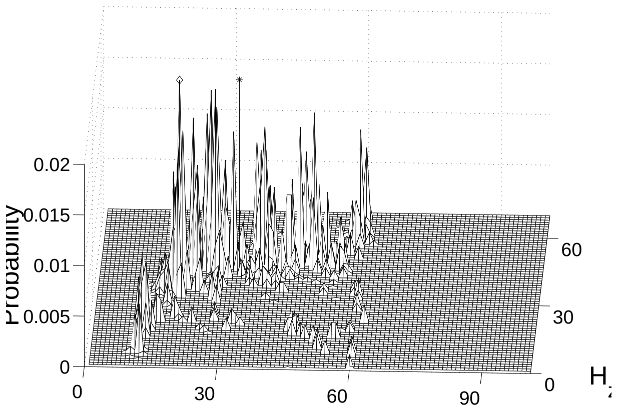

Figure 8. The posterior for figure 3.A. The star illustrate the MAP estimated by a weighted average. The diamond illustrates the MAP estimated by the method proposed in this work.

The method for estimating the MAP that has been suggested in this paper might not always find the highest peak. However, it is always located at a (often large) peak. The standard estimate of the MAP, on the other hand, tends to be located in the centre of the solution space and often at a position where no peaks are present at all, see figure 8. Furthermore, it sometimes happens that the standard estimate of the MAP is located in a pruned region. Obviously, this never occurs when using the method suggested in this paper.

In images such as the one in figure 3.A the posterior is in general ambiguous. In this particular case a correct pose can be found by increasing α as the distance between the hand and camera increases. This tendency can be seen in figure 8. Some of the ambiguity is removed by pruning, however, most of the large peaks in figure 8 could be the correct one. This means that the estimated MAP might be incorrect in this particular image. However, due to the ill-posed nature of the problem this will always be the case independent on the chosen tracking framework. The good thing is that in this tracking framework *the* correct peak is virtual always among the largest peaks and will therefore evolve into the next image, hence the tracking approach handles multi hypothesises.

If the application allows for a delay in the output we can smooth all estimated MAPs over time, e.g. with a Kalman filter, to achieve a more consistent tracking. However, this again introduces the problem of ending up in a location where no peak is present or even worse in a pruned location. Future work will therefore include an investigation of a way to smooth the tracking data based on the r largest peaks in each image instead of only *the* largest peak (found by the estimated MAP) in each image. One approach could be to use an expression similar to 6 and expand it with the weighted displacements over time to force an overall smooth motion on the data.

In this paper we have suggested to apply bootstrapping to increase performance in model-based tracking. Besides the tests presented above the improvement achieved by this approach can also be understood intuitively. Just imagine the complex nature of the posterior having utilising four Euler angles or the screw axis representation without bootstrapping. And the difficulty involved in estimating the MAP. We therefore conclude that bootstrapped tracking is a promising approach when solving some of the inherent problems in model-based tracking.

References

[1] A. Bharatkumar, K. Daigle, M. Pandy, Q. Cai, and J. Aggarwal. Lower Limb Kinematics of Human Walking with the Medial Axis Transformation. In *Workshop on Motion of Non-Rigid and Articulated Objects*, Austin, Texas, USA, 1994.

[2] A. Blake and M. Isard. *Active Contours*. Springer, 1998.

[3] C. Bregler and J. Malik. Tracking People with Twists and Exponential Maps. In *International Conference on Computer Vision and Pattern Recognition*, Santa Barbara, California, June 1998.

[4] A. Hilton. Towards Model-Based Capture of a Persons Shape, Appearance and Motion. In *International Workshop on Modeling People at ICCV'99*, Corfu, Greece, September 1999.

[5] M. Isard and A. Blake. CONDENSATION - conditional density propagation for visual tracking. *International Journal on Computer Vision*, pages 5–28, 1998.

[6] Y. Kameda and M. Minoh. A Human Motion Estimation Method Using 3-Successive Video Frames. In *International Conference on Virtual Systems and Multimedia*, 1996.

[7] T. Moeslund. Pruning the Possible Configurations of a Human Arm using Kinematic Constraints. Technical Report CVMT 01-01, Laboratory of Computer Vision and Media Technology, Aalborg University, Denmark, 2001.

[8] T. Moeslund. Modelling the Human Arm. Technical Report CVMT 02-01, Laboratory of Computer Vision and Media Technology, Aalborg University, Denmark, 2002.

[9] T. Moeslund, M. Vittrup, K. Pedersen, M. Laursen, M. Sørensen, H. Uhrenfeldt, and E. Granum. Estimating the 3D Shoulder Position using Monocular Vision and a Detailed Shoulder Model. In *International Conference on Imaging Science, Systems, and Technology*, Las Vegas, USA, June 24-27 2002.

[10] S. Ross. *Introduction to Probability and Statistics for Engineers and Scientists*. Wiley Series in Probability and Mathematical Statistics, 1987.

[11] D. Tolani and N. Badler. Real-Time Invers Kinematics of the Human Arm. *Presence*, 5(4), 1996.

[12] D. Tolani, A. Goswami, and N. Badler. Real-Time Invers Kinematics Techniques for Anthropomorphic Limbs. *Graphical Models*, 62(5), 2000.

[13] V. Zatsiorsky. *Kinematics of Human Motion*. Champaign, IL: Human Kinetics, 1998.