Pose Estimation by Applied Numerical Techniques

Michael Vittrup, Mads K. D. Sørensen, Brendan McCane

Abstract— **Pose estimation using analysis-by-synthesis techniques has been a topic within computer vision for quite a time. These techniques often make use of numerical optimization algorithms required to work in high dimensional work spaces. In this article we apply two algorithms which to the best of our knowledge have never been used in this context. The first algorithm is based on a generalisation of the snake principle known from active contour, while the second makes use of the simulated annealing principle. In this paper these algorithms are applied to the problem of hand pose estimation, known to be a very difficult task. The methods are based on silhouette matching; matching input silhouettes with synthetically generated silhouettes, and furthermore makes use of stereoscopic vision.**

Keywords— **Numerical Optimisation, Pose Estimation, Analysis-by-Synthesis, Optimisation Algorithms, Downhill Simplex, Simulated Annealing, Spider**

I. Introduction

HAND pose estimation and gesture recognition has been subject to intense research ([1]), as a means of next generation Human-Computer Interaction (HCI). Traditionally, researchers have split approaches into an appearance-based approach and a model-based approach as done by Pavlovic et al. [2].

The appearance-based approach requires a set of predefined gestures, from which some unique image descriptors for the individual gestures are extracted. Extracting the same descriptors from an input image to be processed, a classification routine returns an informed guess of the gesture. Work in this area include Birk et al.[3] and Yonghua [4].

Whereas the appearance-based techniques rely on specific physiological knowledge of the hand, such as the position of the finger tips compared to the wrist and the length of the individual fingers, the model-based approach uses a synthetic model of the hand: A projection of a synthesised model is compared to the input image using some kind of descriptor - overlapping silhouette, contour distance, etc. This comparison is applied in the decision process of how to deform the model in order to bring it into agreement with the input image.

An energy function representing the difference in descriptors between the synthesised model and the input image can be represented using a 23 DoF vector. Thus, the task comes down to minimising the energy function $f(\mathbf{s})$, where $\mathbf{s}$ entirely represents the configuration of the hand. Numerical optimisation techniques can be applied to finding this global minimum which gives the correct pose estimation. These techniques include Downhill Simplex, Levenberg-Marquardt and Powell methods which are discussed and compared in the work by Ouhaddi and Horain [5].

Michael and Mads are with the Computer Vision and Graphics group at Aalborg University, Denmark. E-mail: {babyford,purple}@control.auc.dk .

Brendan is Senior Lecturer at the Computer Science Department, University of Otago, New Zealand. Email: mccane@cs.otago.ac.nz .

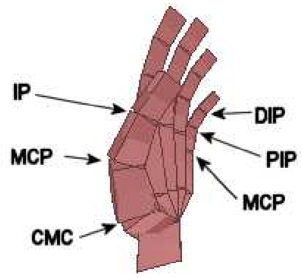


Fig. 1. **The hand model used, with joint abbreviations.**

In this article two new algorithms are proposed, a *Spider* and a *Downhill Simplex/Simulated Annealing* algorithm, which differ widely in their search strategy.

So far, the majority of published work trying to estimate the pose of the hand assumes the wrist to be fixed, either with respect to translation or rotation or both [6], [5], [7], [8]. We experiment with removing these constraints in this work.

In order to test the *algorithms* and not the image segmentation, synthetic input silhouettes are generated instead of using real-life images. Thus, the aim is to fit a model of the hand to a synthetic hand (both rendered using the same model), without knowing the configuration of the latter.

II. Hand Model

THE hand is a very complex structure, but also highly constrained, and many proposals with respect to hand simplifications and constraints have been published; we choose to apply the definitions from Teo[9].

A. Constraints

The joints depicted in figure 1 have one or two DoF each, their static constraints are summarised in table I. *Flexion* and *extension* refer to movement of fingers respectively towards and away from the palm, denoted θ, *abduction* and *adduction* refer to movement of finger away from or closer to the plane dividing the middle and ring finger (perpendicular to the palm), referred to as δ. This yields a total of 27 DoF including wrist translation and rotation. Rotation of the wrist is restricted to lie within $\angle X \in [0; 180]$; $\angle Y \in [-90; 90]$ and $\angle Z \in [-180; 180]$, translation along each axis within 5 centimetres. These constraints are based on the specific Watching Window [10] console, for which this system is intended. These constraints and the constraints in table I narrow the search space, but by utilising *dynamic* constraints, the search space can be narrowed even more.

TABLE I
Static constraints of each joint.

Joint	DoF	Flexion / extension (θ)	Abduction / adduction (δ)
DIP	1	60	–
PIP	1	100	–
MCP	2	90	60
IP	1	85	–
MCP	2	50	30
CMC	2	120	45

Vardy [11] presented an intuitive constraint; when flexing the PIPs, the corresponding DIP on the same finger will automatically flex as well. This relationship can be approximated to:

$$\theta_{DIP} = \frac{\theta_{PIP}}{3} \tag{1}$$

This simple formula narrows the search space by four DoF. Another coherence is the dependence of maximum and minimum δ-values as function of the level of extension. If the finger is fully extended, the whole range of δ can be utilised, if the finger is flexed, δ will be significantly restricted. In [11] this relationship is approximated by a simple formula:

$$\delta_{MCP} = \frac{\theta_{MCP} \times 15^o}{90^o} - 15^o \tag{2}$$

δ_{MCP} is the maximum level for δ for each finger for this specific model.

III. Cost function

The cost function used to evaluate the correspondence between the input images and the synthetic hand pose must be defined. In this work we consider the surface difference between the silhouette of the hand in the input images and the evaluation silhouette synthesised, computed for each viewpoint and summed. Adopting the cost function from Kameda [8] we define an "outer pressure condition":

$$f_{out}(\mathbf{s}) = \sum_{\mathbf{i} \in D} M(\mathbf{i}) \cdot (1 - S(\mathbf{i})) \tag{3}$$

and an "inner pressure condition":

$$f_{in}(\mathbf{s}) = \sum_{\mathbf{i} \in D} S(\mathbf{i}) \cdot (1 - M(\mathbf{i})) \tag{4}$$

where S is the input silhouette image, M is the evaluation silhouette image (both assumed binary) and D represents the entire image. M is synthesised from the pose vector to be estimated, $\mathbf{s}$. Minimising f_{out} causes the evaluation silhouette(M) to be fitted within the input silhouette(S), while minimising f_{in} helps fitting the input silhouette (S) inside the evaluation silhouette (M). By combining these into one equation,

$$f(\mathbf{s}) = f_{in}(\mathbf{s}) + f_{out}(\mathbf{s}) \tag{5}$$

we obtain a function which punishes any difference between the input and evaluation silhouettes.

The magnitude of f, the *silhouette error*, depends on the area of the silhouettes, thus different scaling situations yield different magnitudes of f. As we allow for wrist transformation this is unwanted and thus we propose and use a normalisation of equation 5, yielding equation 6.

$$f(\mathbf{s}) = \frac{f_{in} + f_{out}}{\sum_{i \in D}(S(i) + M(i))} \tag{6}$$

The function, f, is a reasonable measure for estimating the unknown pose of a hand, however, it is not adequate for evaluating how good that estimation is. For example, a hand facing palm forwards compared to one facing palm backwards will produce a small silhouette error, yet be a poor estimate of the pose of the hand. So, for evaluation purposes, since we know the actual pose of both the model and the hand we are trying to estimate (since it too is synthetic), we define the pose error as:

$$P = \sum_{i=0}^{22} \sqrt{((x_i^{in} - x_i^{eval})^2 + (y_i^{in} - y_i^{eval})^2 + (z_i^{in} - z_i^{eval})^2)} \tag{7}$$

x_i^{in} is the spatial x-position of the known ith joint, x_i^{eval} is the estimated x-position of the ith joint and so forth.

IV. Optimisation Algorithms

The aim of analysis-by-synthesis is to minimise the cost function presented in the previous section. To perform this minimisation we propose two algorithms with very different characteristics. Both use the pose estimation vector from the previous frame as the starting vector for the next frame.

A. *The Spider Algorithm*

The spider algorithm, a generalisation of the snake principle commonly used in active contours, is due to McCane [12].

A.1 Representation

The hand is represented in a tree structure as shown in figure 2. For finger joints, 1D nodes for representing one joint angle are used, i.e. δ for the MCP joint for the little finger. Each of these nodes holds three possible values to be tested. The values in the nodes are initiated using the latest pose estimate, $\mathbf{s}$. One value is assigned the value for the previous pose estimation, and the other two an increment and decrement of the individual joints, respectively. So if the possible values for joint i are to be determined the values will be S_i and $S_i \pm \Delta_i$, thereby searching the vicinity of the previous pose estimation. For the wrist position and rotation, 3D nodes (cubes) are used for the representation. A cube is a generalisation of the 1D node to three dimensions. Each cell in a cube holds set of values to be tested. In the node representing wrist translation for instance, a cell contains a (x, y, z) set to be tested. By using 3D nodes for the wrist position we obtain more focus on their interrelationship.

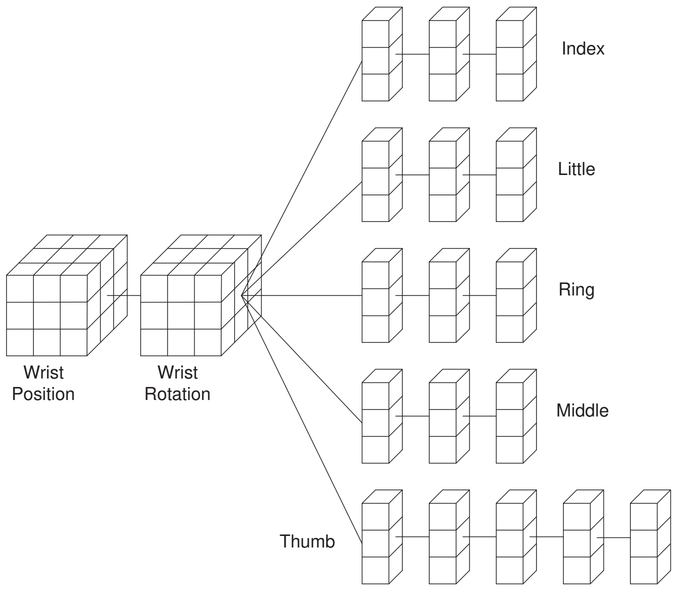


Fig. 2. Spider Tree

A.2 Step Size

Choosing the correct Δ_i, which we denote "step size", for each joint is very important for the performance of the spider, as these values decide how close the exact pose of the hand one can expect to come. One can think of the step size as the resolution of the spider; if large Δ_i value, say 5, is chosen one cannot expect to estimate the pose with a precision of more than 5 degrees. On the other hand the spider might avoid some of the local minima. Choosing a low value, i.e. $\Delta_i = 0.1$ enhances the precision of the final estimate, but the spider will require a large amount of iterations to obtain a minimum energy and will be prone to getting caught in local minima.

We choose to gradually decrease the Δ values, thus starting out with searching a big neighbourhood of the previous pose, and gradually refining the search. In this study the step size is decreased linearly from the values in table II, but other schedules, i.e. exponential, could be subject to further investigation. Because changes in wrist position and rotation usually results in larger silhouette changes than a similar change of a finger joint, the initial step sizes are different and heuristically chosen.

TABLE II
Initial step sizes

Joints	Step Size
Wrist position	1
Wrist rotation	7
Finger joints	10

A.3 Minimisation

Dynamic programming is applied to evaluate the energy of the nodes in the tree structure in a similar manner to minimising a snake using dynamic programming:

$$S(v_p) = \sum_{v_c \in C(v_p)} min_{v_c}(S(v_c) + E_{int}(v_p, v_c) + E_{ext}(v_p, v_c)) \tag{8}$$

Here v_p and v_c are the parent and child node respectively. $S(v_p)$ is the final energy of the parent node.

This is a modification from the original spider, as we make the external energy dependent on the parent/child relationship.

A.4 Energies

An external E_{ext} and internal E_{int} energy function have to be provided to evaluate equation 8. Here we choose the external energy to be the cost function proposed in equation 6. The pose vector, $\mathbf{s}_{init}$, estimated from the previous frame is copied to $\mathbf{s}'$, and the values of v_p and v_c currently being evaluated are then inserted into the correct positions in $\mathbf{s}'$. We then set $E_{ext} = f(\mathbf{s}')$.

The internal energy E_{int} can be used to favour different hand poses, for instance favouring relaxed poses. In this feasibility study, however, the internal energy is set to zero.

B. Downhill Simplex/Simulated Annealing

A different multidimensional search algorithm to be used in hand pose estimation, Downhill Simplex (DS), is due to [13] and was applied to hand pose estimation by [5], though they did not take wrist rotation into account. By improving DS with Simulated Annealing (SA), full spatial motion will hopefully be allowed for. Our implementations are derived from [14].

B.1 Downhill Simplex

A simplex is the most elementary geometrical figure that can be formed in $\Re^n$ and has $n+1$ vertices (each vertex is a n-dimensional vector representing a point in $\Re^n$), so the simplex consists of a series of states, $\mathbf{s_0} \ldots \mathbf{s_n}$. On a plane (two dimensions, $\Re^2$), a simplex is a triangle. In 3D space, $\Re^3$, a tetrahedron, in higher dimensions a hyper tetrahedron. If any vertex of a simplex is considered as the origin, the n other vertices define vector directions spanning all of $\Re^n$.

Only one vertex is specified, $\mathbf{s}_{init}$, the other n vertices are found using equation 9 where $\mathbf{e_m}$ is the m'th unit vector and λ_m specifies the length scale of this vector direction, specified to maximise the simplex in that direction; though still within the constraints.

$$\mathbf{s}_m = \mathbf{s}_{init} + \lambda_m \mathbf{e}_m \tag{9}$$

for $m = 0 \ldots n-1$. A simple choice for $\mathbf{s}_{init}$ is the pose estimate from last frame; the initial simplex is built up from this vertex and the axes of $\Re^n$ are added.

Several hundred transformations of the simplex are now performed, each transformation generating a new simplex based on its previous geometry and the function evaluations at every vertex. The transformations can be one of the transformations depicted in figure 3 (in $\Re^2$, for clarity reasons).

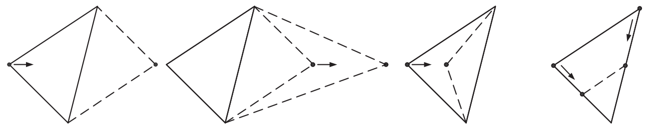


Fig. 3. DS geometrical transformations in $\Re^2$

	Silhouette Error			Pose Error		
Seq	DS	Spider	DS/SA	DS	Spider	DS/SA
1	0.0129	0.0101	0.0179	12	14	18
2	0.0114	0.0058	0.0231	18	10	36
3	0.0158	0.0103	0.0196	15	17	21
4	0.0553	0.0049	0.0601	67	9	83
5	0.0108	0.0058	0.0148	16	11	16
6	0.0144	0.0064	0.0199	17	10	22

TABLE III
Error table, fixed wrist

The exact order of transformations depends on the energy evaluations at each vertex, but the bottom line is that an appropriate sequence of these steps will always converge to at least a local minimum energy vertex of the specified objective function.

B.2 Simulated Annealing

The theory of simulated annealing (SA) applies thermodynamic physical laws to optimisation problems. When liquids freeze and crystallise the molecules start with high thermal mobility, freely moving among another. As the temperature slowly decreases this mobility is gradually lost and the molecules form a pure, highly ordered crystal; known as the minimal energy state of the system.

The Metropolis Algorithm ([15]) simulates a thermodynamic system by letting the search go downhill if an option, but *sometimes* go uphill, allowing it to escape from local minima.

The probability of going uphill (p) can, bypassing the physics, be expressed using equation 10:

$$p = e^{\frac{-(E_{old}-E_{new})}{T}} = e^{\frac{-\Delta E}{T}} \tag{10}$$

where T symbolises the temperature, E_{old} and E_{new} the cost function evaluated at the old and the new point in $\Re^n$, respectively. If $\Delta E < 0$, p is arbitrarily set to 1. Thus, the probability of accepting the new point is p.

The Metropolis Algorithm is in [14] implemented by modifying the basic DS method by *adding* a logarithmically distributed random variable based on T to the energy value E associated with every vertex of the simplex, and *subtracting* a similar random value from the energy value of any new vertex tried as a replacement point.

Using this method, a downhill step is always accepted, but uphill steps occur occasionally. For $T \to 0$ the influence of the stochastic element will be infinitesimal and the algorithm will reduce to the basic greedy DS algorithm for $T = 0$, thus converging to a local minimum.

In this specific case, 2,000 evaluations are allowed, a linear cooling schedule is chosen, and a starting temperature of $T_{init} = 0.02$, an end temperature of $T_{end} = 0.00$ distributed over eight temperature intervals. This combination seems to yield an acceptable ratio of performance versus computational burden.

V. Results

The three algorithms presented are tested to evaluate their performance. A series of twelve sequences are designed using eight predefined gestures, such as a fist, a pointing finger, the victory sign etc. Every sequence consists of 250 frames in which morphing between the gestures is performed. In six of these sequences we prohibit the wrist from moving, while the last six sequences features a moving wrist.

The basic DS method is used as a standard of reference, as it has already been used for pose estimation [5]. The DS is allowed one restart in each frame as initial tests showed the DS to perform better if this is allowed.

A. Fixed Wrist

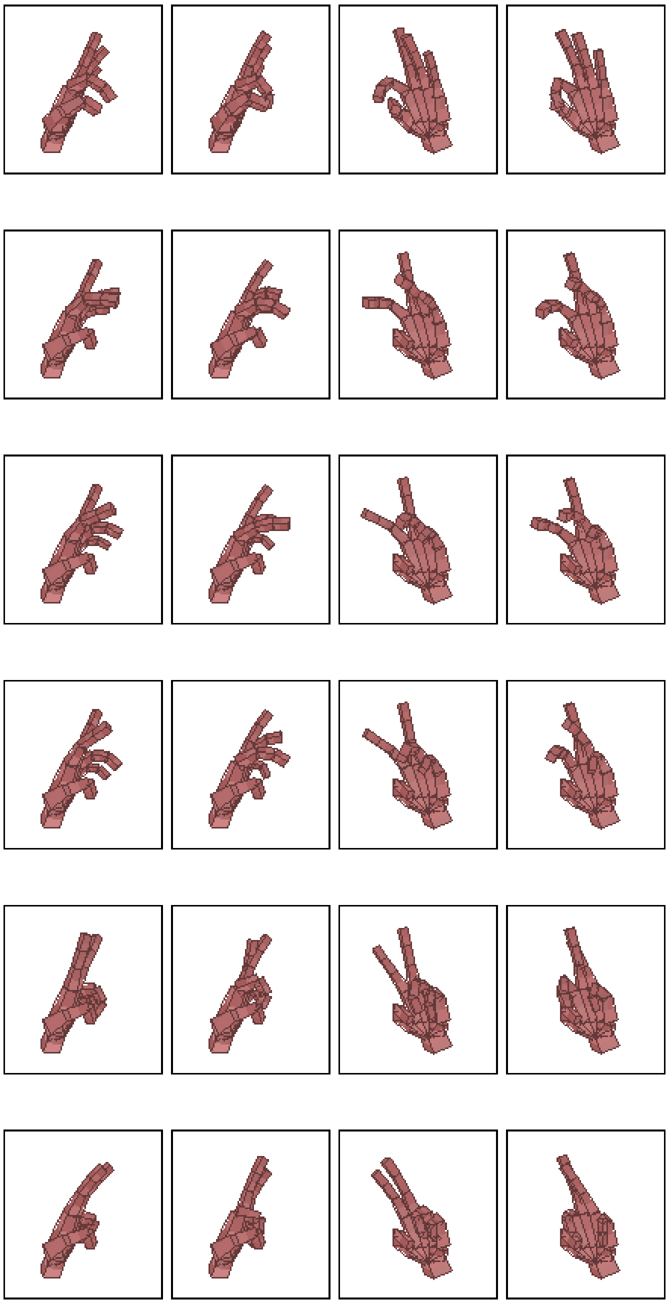
In sequences where the wrist is fixed, all three algorithms deliver impressive estimations. In general, flexion/extension and abduction/adduction of all fingers are successfully detected by the algorithms, with a few exceptions. An example of such an exception is shown in figure 4.

The problems typically arise when the estimation has to be based on one input image. As can be seen in the figure, the left input image contains a lot of ambiguity, thus the estimation should theoretically be based on the right input image only.

The Spider algorithm performs just as good as the DS algorithm on these tests, and outperforms both in estimating sequence 4. Interestingly enough the DS performs better on average than DS/SA on these sequences, although they both perform to an acceptable level. The reason may be that the parameters of DS/SA are adjusted to sequences with non-fixed wrist, and that there is nothing to optimise that the DS not already do very well. The conclusion is that adding randomness to cases where an algorithm already performs well can do more damage than good.

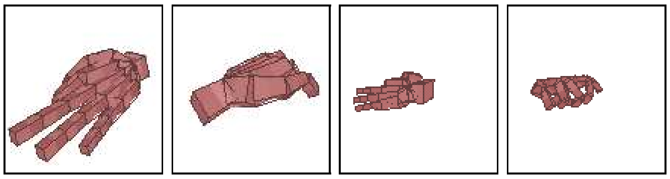
B. Non-fixed Wrist

Allowing for wrist translation and rotation, the situation changes. Often the algorithms perform very well, capable of estimating the correct wrist transformation. However, due to the ambiguous silhouettes, all optimisation methods show problems at times in this case; occasionally estimating the wrist rotation 180 degrees wrong, yielding a low



(u) Left input (v) Left estimate (w) Right input (x) Right estimate

Fig. 4. **Example of estimation failure. (u) and (w) show the input images to the silhouette fitting, while (v) and (x) show the estimated poses. Shown are frames 279, 282, 284, 285, 290 and 300.**



(a) Left input (b) Left estimate (c) Right input (d) Right estimate

Fig. 5. **Example of the wrongly estimated pose, showing how the hand orientation is estimated approximately** 180^o **wrong.**

	Silhouette Error			Pose Error		
Seq	DS	Spider	DS/SA	DS	Spider	DS/SA
1	0.0528	0.0825	0.0650	64	76	104
2	0.0980	0.0605	0.0706	115	48	113
3	0.0652	0.0648	0.0420	73	63	39
4	0.0891	0.0809	0.0726	126	104	99
5	0.0856	0.1098	0.0739	107	131	91
6	0.0760	0.0896	0.0632	114	100	76

TABLE IV

Error table, non-fixed wrist

silhouette error $f(\mathbf{s})$ but a high pose error P; see table IV. An example is shown in figure 5.

On the average DS/SA and the Spider method comes out on top, with both performing well for a significant amount of the time. However, all three algorithms in general seem to accumulate errors over time. An example of this is shown in figure 6, visualising the pose error for the same sequence over time for the three algorithms.

In this graph we notice how all three algorithms perform with a very low pose error the first 70-80 frames. After this the pose error of all three algorithms rise to between 150

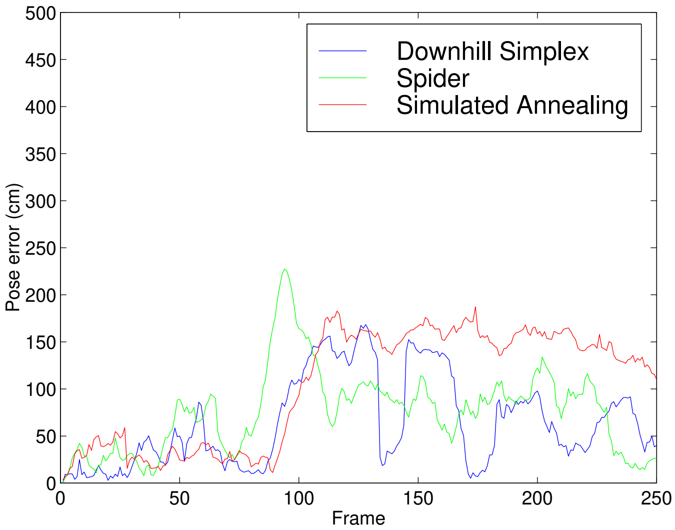


Fig. 6. **Example of how errors get accumulated over time.**

and 220. Notice how all three algorithms have difficulties obtaining the right pose after this. While the Spider manages to bring the pose error down, it still levels at around 90, caught in a local minimum. Likewise the DS/SA has big problems, and actually never recovers from the rise. The DS though seems at times capable of performing very precise pose estimations, as for instance around frame 175 where the pose error is around 20 cm only to rise to 100 cm over a span of 10 frames. The same pattern is observed around frame 140. This is due to the ambiguity of the pose measurement where the hand is actually rotated 180^0 around its z-axis. While estimating different gestures result in high pose errors, an open hand, although rotated 180^0 gives a low silhouette error.

The overall conclusion to the test is that all three algorithms delivers a good pose estimation when the wrist is fixed. Allowing for wrist transformation, however, the algorithms will always fail at some point.

VI. Conclusion

In this article we have introduced two new algorithms to be used for analysis-by-synthesis optimisation problems. In this case the algorithms were applied to the problem of hand pose estimation by silhouette matching. The proposed algorithms show good estimation performance if the wrist is fixed, but perform less well when global translation and rotation are introduced. This is caused by the ambiguity of the silhouettes and with it an ambiguous cost function.

The problem of hand pose estimation is known to be a very difficult problem due to the hands flexibility. The two introduced algorithms perform with varied success in this case, but may prove successful in problems with a lower dimensionality.

VII. Future Work

To accomplish the task of hand pose estimation, it is our belief that other techniques must be applied to complement the proposed algorithms - especially estimating the wrist using an appearance-based approach *before* adjusting the fingers using this model-based technique is necessary, if a stable solution is to be reached.

The presented algorithms do not work in real time. The present implementation makes use of hardware acceleration to synthesise the silhouettes. Whilst very efficient for drawing images to the screen, it seems it is much less efficient to transfer those images to main memory for further processing. Our experiments indicate the machine is capable of performing only 50 function evaluations per second (transferring the image and calculating the silhouette differences). Since our hand model is made up of simple parts and we are only interested in the silhouette, we believe we could achieve significant speedup by using a specialised software renderer. It is unclear however, if this would allow us to achieve real-time performance and this is an area for future work.

References

[1] T. Moeslund and E. Granum, "A survey of computer vision-based human motion capture," *Computer Vision and Image Understanding: CVIU*, vol. 81, no. 3, pp 231–268, 2001.

[2] Vladimir Pavlovic, Rajeev Sharma, and Thomas S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, 1997.

[3] Henrik Birk, Thomas B. Moeslund, and Claus B. Madsen, "Real-time recognition of hand alphabet gestures using principal component analysis," *Proceedings of the 10th Scandinavian Conference on Image Analysis*, 1997.

[4] Dong Guo Yonghua, "Vision-based hand gesture recognition for human-vehicle interaction," *International Conference on Control, Automation and Computer Vision*, 1998.

[5] Hocine Ouhaddi and Patrick Horain, "Hand tracking by 3d model registration," in *Actes du Colloque Scientifique International Realite virtuelle et prototypage*, 1999, pp 51-59.

[6] James M. Rehg and Takeo Kanade, "Visual tracking of high DOF articulated structures: an application to human hand tracking," in *ECCV (2)*, 1994, pp. 35–46.

[7] James Davis and Mubarak Shah, "Toward 3-d gesture recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 13, no. 3, pp. 381-393, 1999.

[8] Yoshinari Kameda, M. Mino and K. Ikeda, "Human pose and motion estimation with a three dimensional articulated object model," in *Asian Conference on Computer Vision*, 1993.

[9] Chor Guan Teo, "A hybrid procedural/knowledge-based approach to the animation of human hand grasping," *Masters Thesis, Simon University*, 1994.

[10] Alister West and Keekim Heng, "The watching window: A reactive real-time model-viewer," in *Image and Vision Computing New Zealand*, 2001, pp. 25–30.

[11] A. Vardy, "Articulated human hand model with inter-joint dependency constraints," `http://www.scs.carleton.ca/~avardy/hand/hand_doc.pdf`, 1998.

[12] Brendan McCane, "Snakes and spiders," in *Proceedings of International Conference on Pattern Recognition* , pp. 652-655, Barcelona, Spain, 2000.

[13] J.A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.

[14] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, *Numerical Recipes in C, Second edition*, Cambridge University Press, 1992.

[15] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *Journal of Chemical Physics*, pp. 1087–1092, 1953.