



# Silhouette Area Based Similarity Measure for Template Matching in Constant Time

Daniel Mohr

Clausthal University, Germany

[dmoh@tu-clausthal.de](mailto:dmoh@tu-clausthal.de)

*AMDO 2010, Andratx, Mallorca, Spain*



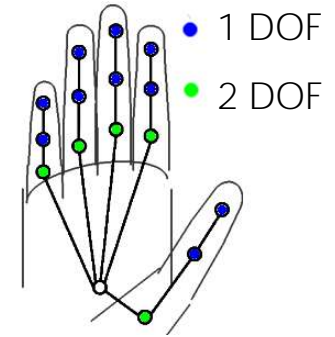
# Motivation: Camera Based Hand Tracking

- Given an image, estimate hand parameter

- Global position (3 DOF)
- Global orientation (3 DOF)
- Joint angles (20 DOF)



global state



local state

- Tracking approach

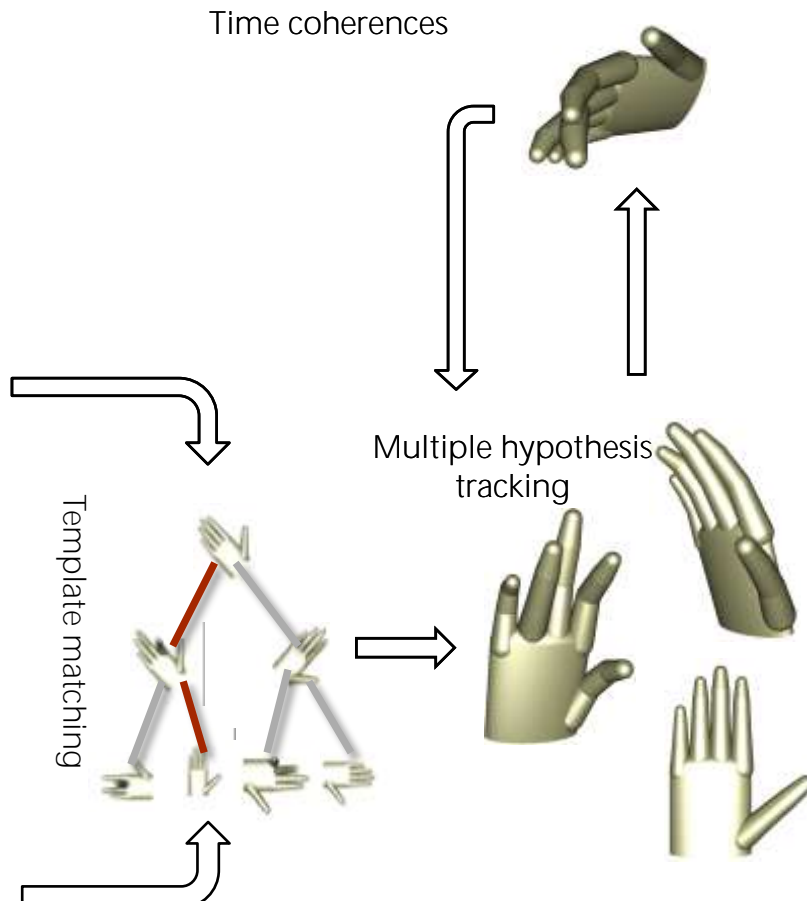
- Sample hand parameter space  $\theta_1, \dots, \theta_N, \theta_i \in \mathbb{R}^{26}$
- Project hand models onto 2D and compare with query image
- Estimate global position by position/scale of the hand in the query image and orientation/joint angles by different templates



# Hand Tracking Pipeline



Edge Feature  
Region Based Feature





# Template Matching



	Generate templates on-the-fly	Precompute templates
Number of templates	unlimited	limited to precomputed poses
Storage space	constant	linear in #templates
Matching time	high	low
Additional structures (e.g. hierarchy)	almost impossible	possible
Appropriate for	local search	global search e.g. (re-)initialization



# Related Work



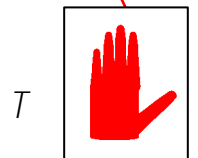
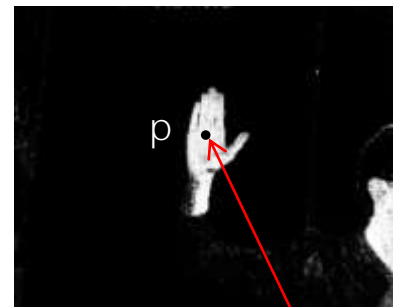
- Binary-Binary matching: template and input image segmentation are binary
  - Direct comparison of foreground regions:
    - Intersection between template and input image segmentation [*Lin et. al AFGR2004*][*Kato et. al AFGR2006*][*Ouhaddi et. al 1999*]
  - Extract higher level features
    - Compare difference vectors between gravity center and points at silhouette contour [*Amai et. al AFGR2004*][*Shimada et. al ICCV2001*]
- Binary-Scalar matching: binary template, scalar segmentation
  - Joint probability [*Stenger et. al PAMI2006*][*Sudderth et. al CVPR2004*]
  - Efficient computation through prefix sum for each line in segmentation [*Stenger et. al PAMI2006*]



# Joint Probability

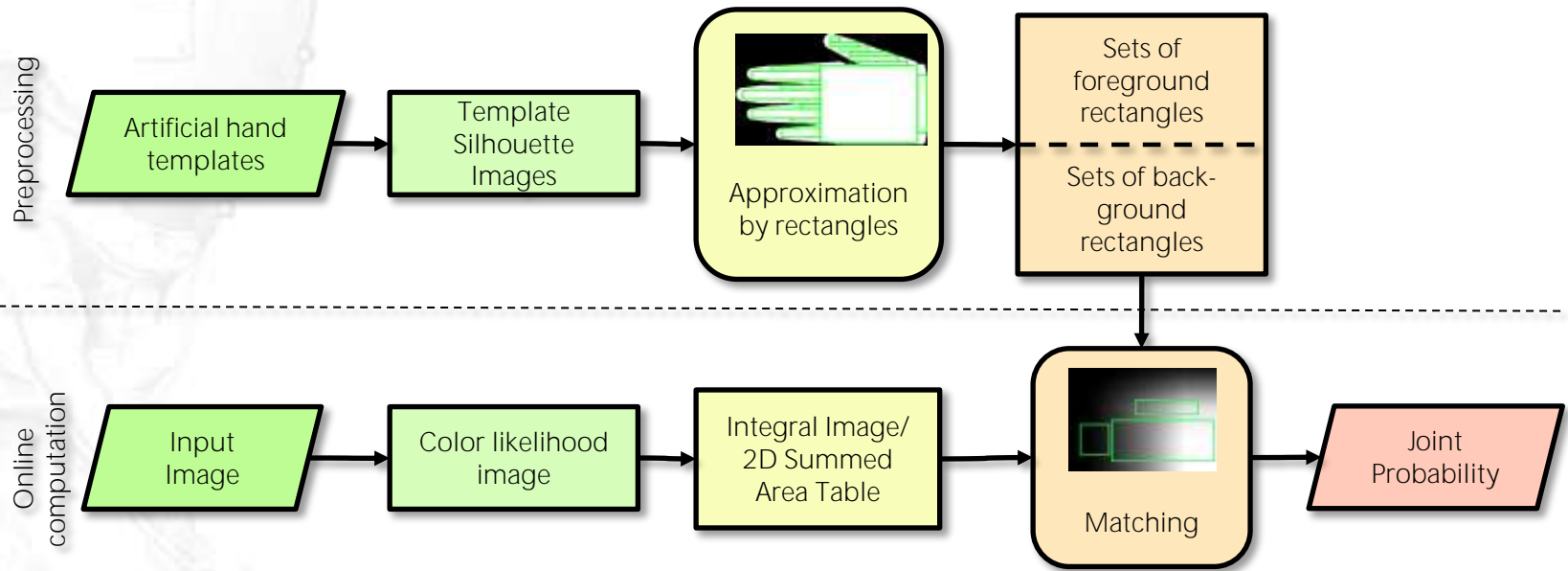
- Given
  - Input image
  - Position  $p$
  - Template  $T$
- Foreground segmentation  $S$  (we use skin color)
- Similarity measure given by joint probability between  $T$  and  $S(p)$

$$P(S, \mathbf{p}, T) = \prod_{\mathbf{x} \in \text{fg}(T)} S(\mathbf{p} + \mathbf{x}) \cdot \prod_{\mathbf{x} \in \text{bg}(T)} \neg S(\mathbf{p} + \mathbf{x})$$





# Fast Area Based Template Matching





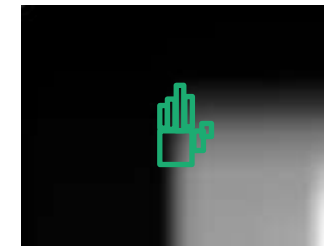
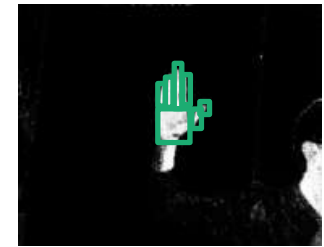
- Preprocess input image Segmentation :
  1. Take logarithm of segmentation S
  2. Compute 2D summed area table IS of log-image
- Use rectangular representation R of template T
  - Joint probability for a rectangle

$$h(R_i) = \sum_{x \in R_i} \log S(x) = IS(R_i.lower.right) + IS(R_i.upper.left) - IS(R_i.lower.left) - IS(R_i.upper.right)$$

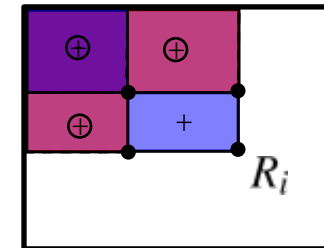
Computation cost per rectangle: 4 look-ups in IS

- Joint probability

$$P(S, \mathbf{p}, T) \approx P(IS, \mathbf{p}, R^*) = \exp\left(\sum_{R_i \in R^*} h(R_i + \mathbf{p})\right)$$



(0,0)

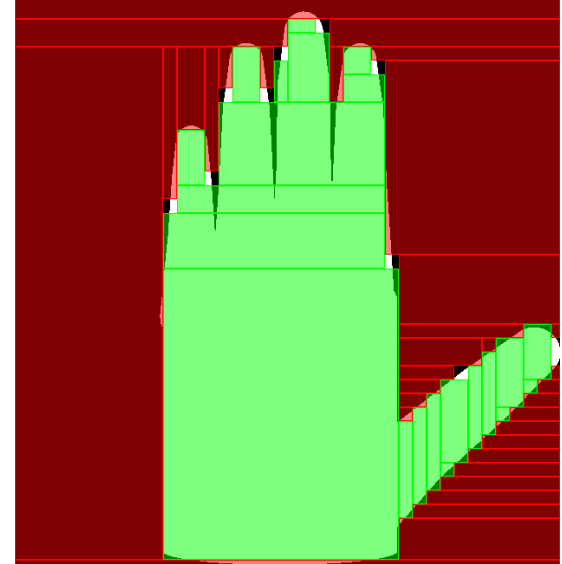






# Template Representation Computation

- For all templates  $T$ 
  - Approximate foreground/background area by a set  $R$  of **axis-aligned rectangles**
- Criteria for rectangle covering
  1. Cover as much area as possible (param  $\tau$ )
    - High matching accuracy
  2. Use as few rectangles as possible (param  $\theta$ )
    - Faster matching
    - Less memory consumed by template
- Define benefit function



$$g(R_i) = -\theta + \sum_{(x,y) \in R_i} (T(x,y) - \tau), \quad R_i \in R$$



# Our Rectangle Covering Algorithm Computation

- Optimization Function

$$R^* = \operatorname{argmax}_{R \subset \mathcal{R}} \sum_{R_i \in R} g(R_i)$$

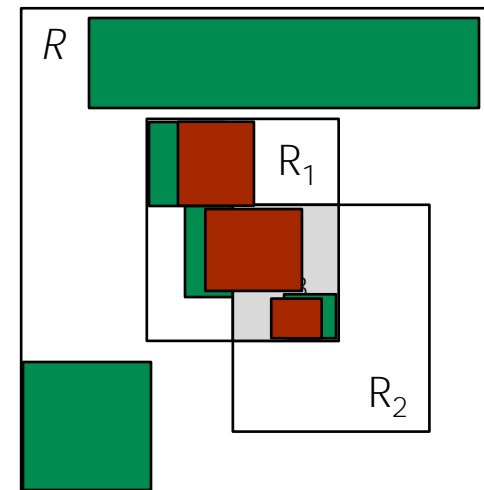
- Solve our rectangle covering problem by dynamic-programming

- *Optimal substructure property:*

- Let  $R_1^*$  be the optimal solution for a rectangle  $R_1 \subset R$
- If  $R_1$  or any subset is in the optimal solution  $R^*$  of  $R$ , then  $R_1^* \subset R^*$

- *Overlapping subproblems*

- $R_3 = R_1 \cap R_2$  is needed to computing the optimal solution of  $R_1$  and  $R_2$

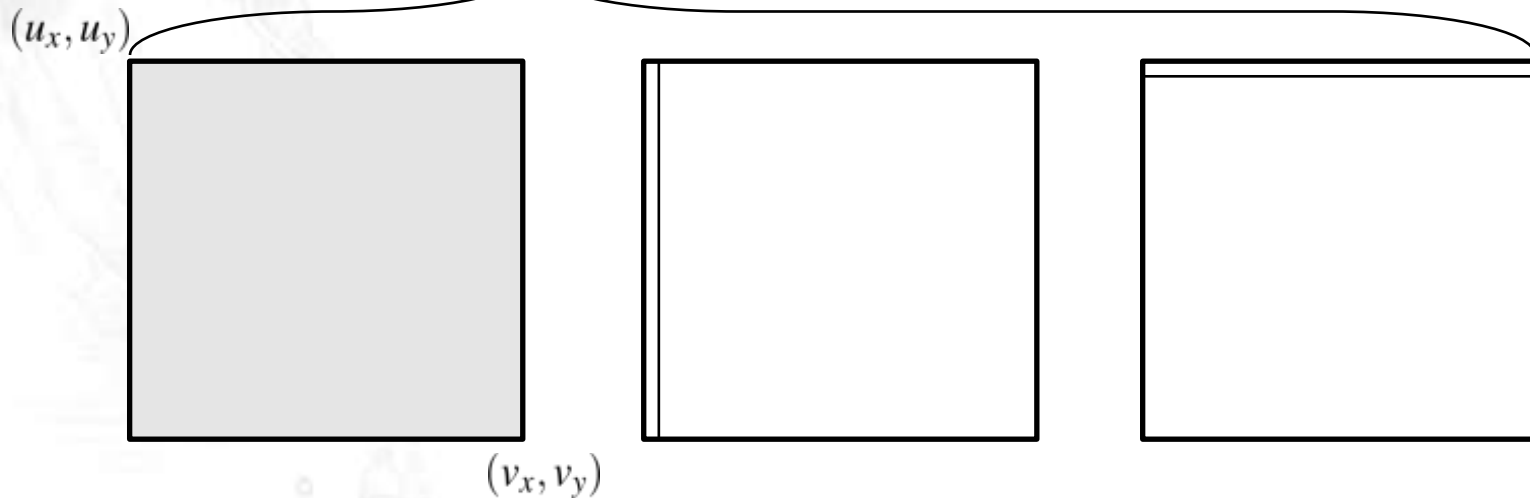




# Template Representation Computation

- Recursive equation

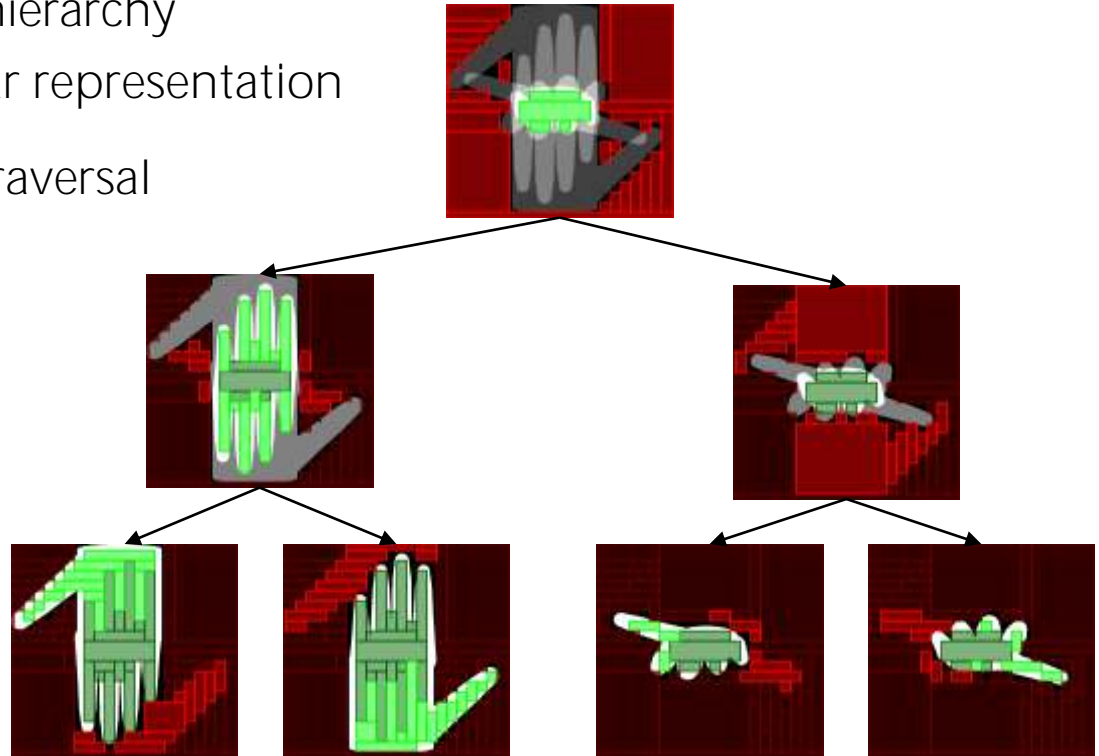
$$\mathcal{D}(R_{v_x, v_y}^{u_x, u_y}) = \max \left\{ \begin{aligned} &g(R_{v_x, v_y}^{u_x, u_y}), \\ &\max_{u_x < x < v_x} \left\{ \mathcal{D}(R_{x, v_y}^{u_x, u_y}) + \mathcal{D}(R_{v_x, v_y}^{x, u_y}) \right\}, \\ &\max_{v_x < y < v_y} \left\{ \mathcal{D}(R_{v_x, y}^{u_x, u_y}) + \mathcal{D}(R_{v_x, v_y}^{u_x, y}) \right\} \end{aligned} \right\}$$





# Template Hierarchy

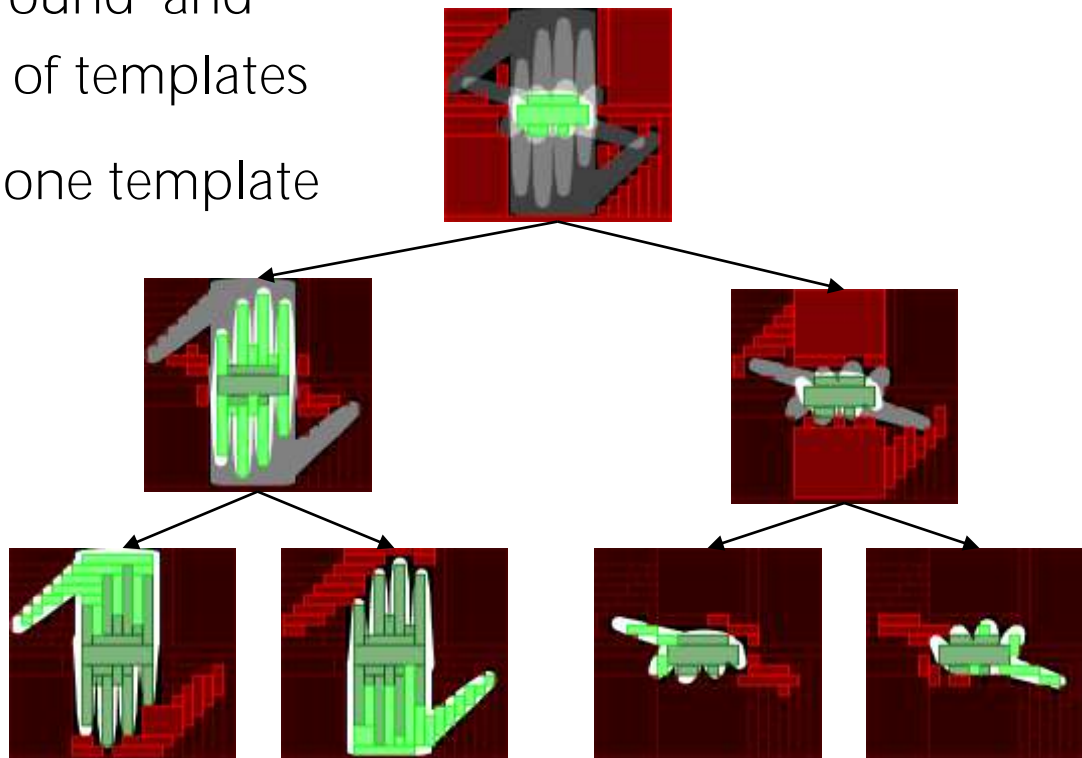
- Match a set of  $n$  templates at a large number of positions in the input image
- Hierarchical approach
  - Generate template hierarchy based on rectangular representation
  - Matching through traversal
  - Complexity reduced from  $O(n)$  to  $O(\log n)$





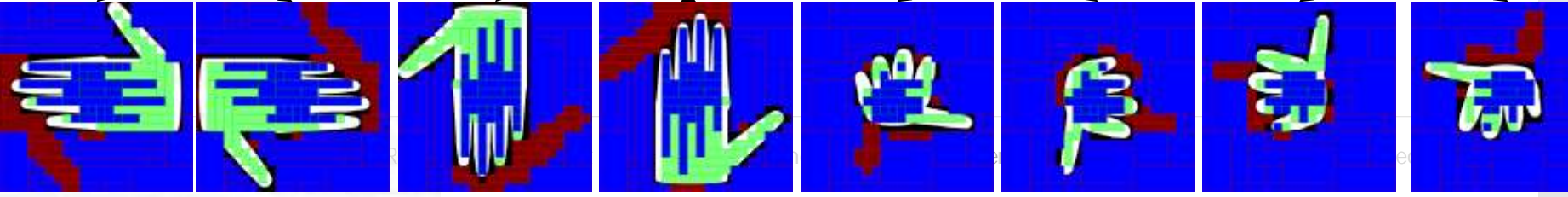
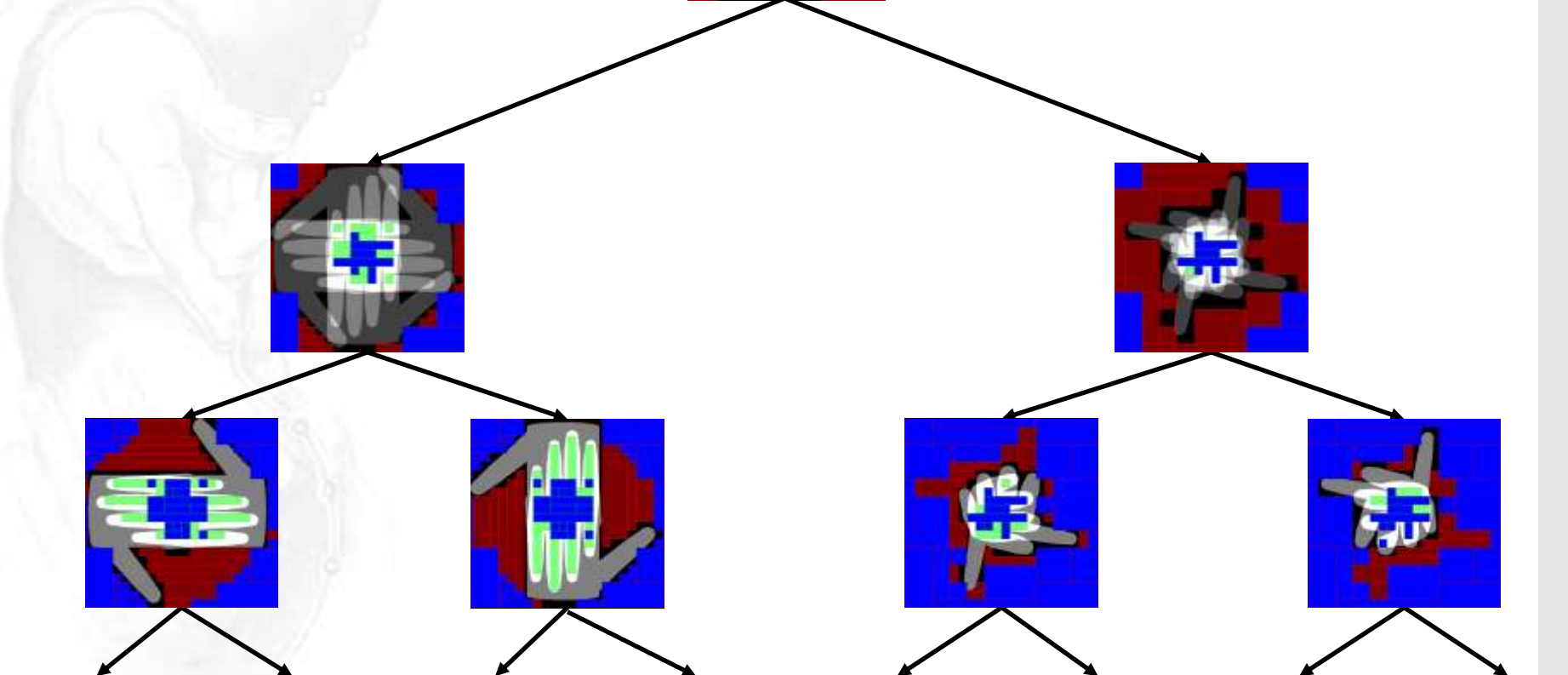
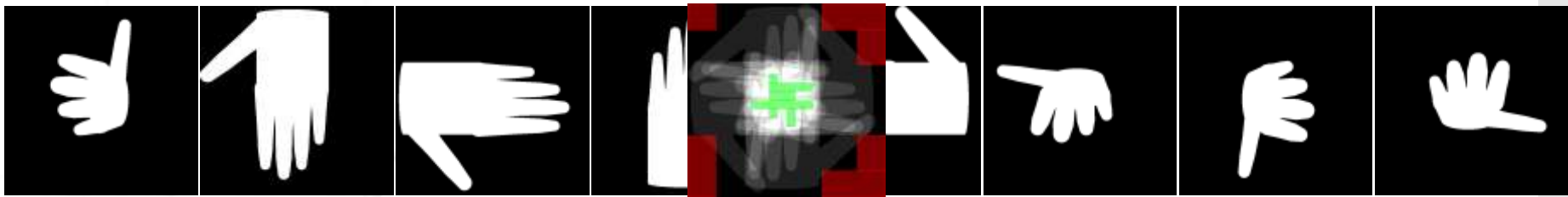
# Hierarchy Generation

- Templates with similar shapes should end up in the same subtree
- Each node contains a set of axis-aligned rectangles that represent the foreground and background regions of templates
- Each leaf represents one template





# Hierarchy Generation: Algorithm





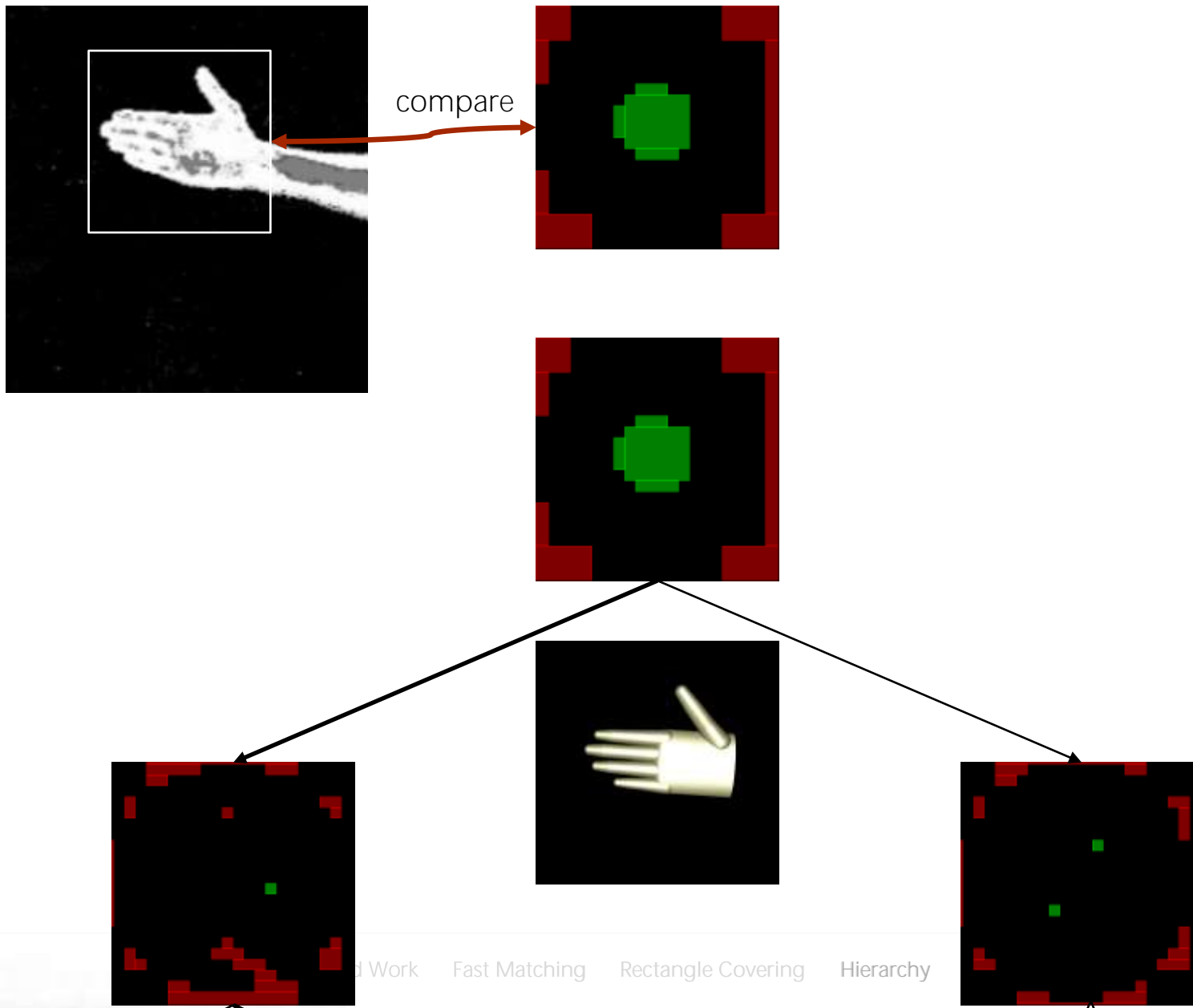
# Hierarchical Matching



1. Start at root node
2. Compute joint probability of regions stored in current node
  - Areas in the template bounding box not yet matched are treated with probability 0.5 (i.e. foreground and background have same probability)
  - Ensure non-decreasing probabilities while moving along a path to a well matching template
3. Compute joint probability at all child nodes
4. Visit child with highest matching probability
  - Multi-hypothesis tracking: follow  $n$  instead of 1 path during traversal
5. If “is inner node” Goto step 2 else finished



# Hierarchical Matching

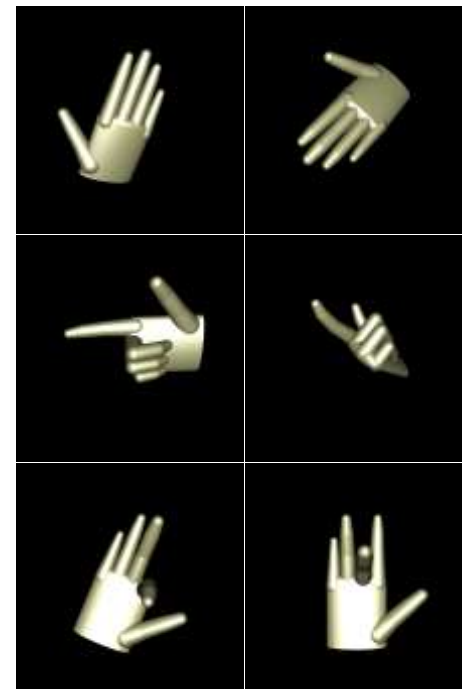






# Experimental Results

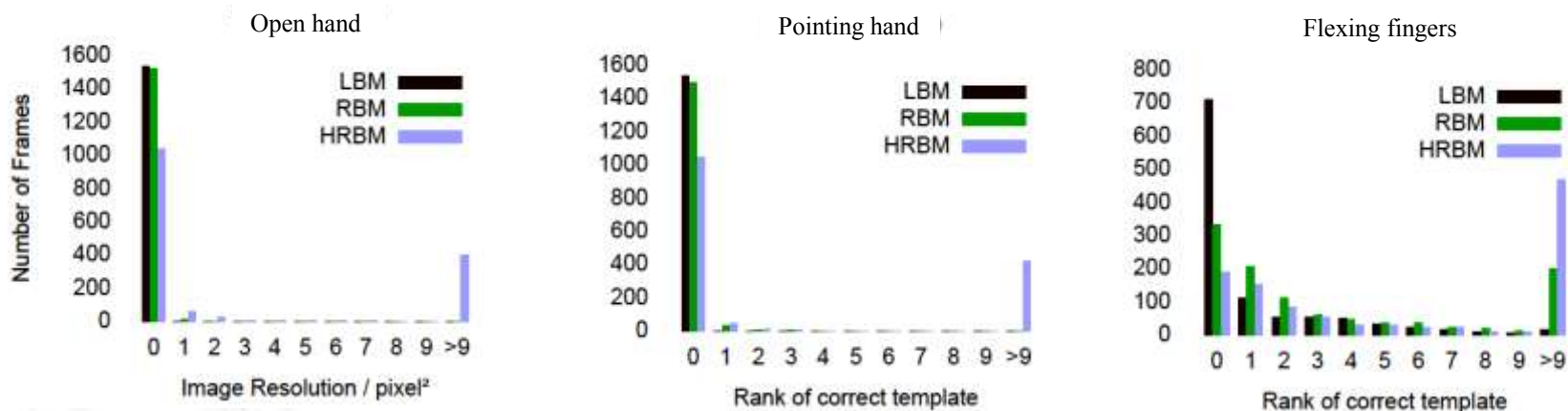
- We use the templates itself as input images because
  - We **compare distance measures** and not full tracker approaches and thus disturbing factors like bad illumination, segmentation noise, varying hand shapes are undesired
  - Ground truth available
- Three datasets
  - **Open hand** (2 rotational DOF)
    - 1536 templates
  - **Pointing hand** (2 rotational DOF)
    - 1536 templates
  - **Flexing fingers** (flex fingers, 1 rotational DOF)
    - 1080 templates





# Matching quality

- The ranks 0-9 of the best matching template are recorded
- The plot shows the rank-histogram for a set of input image



**LBM:** Stengers approach using prefix sum per line

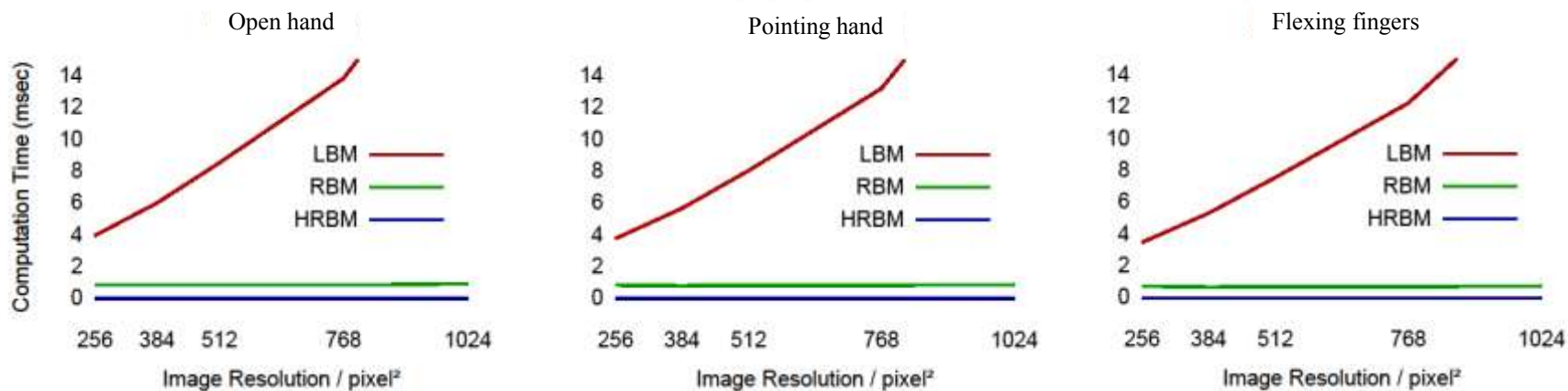
**RBM:** Our approach with axis-aligned rectangles

**HRBM:** RBM using our hierarchy



# Computation time

- Average computation time for each template set



- A resolution of 1024x1024 our approach (RBM) is about **25 times** faster than Stengers method (LBM)

**LBM:** Stengers approach using prefix sum per line

**RBM:** Our approach with axis-aligned rectangles

**HRBM:** RBM using our hierarchy



# Conclusions

- Novel template representation
  - Resolution-independent
  - Low memory cost ( $\sim 0.6$  KByte / *template*)
- Matching is fast and resolution-independent ( $0.7\mu\text{s}$  / *template*)
- Template hierarchy with hierarchical matching
  - Complexity  $O(\log \#templates)$  ( $28\mu\text{s}$  to traverse a tree with  $\sim 1500$  *templates*)
  - Hierarchy offers time critical matching: accuracy can be chosen online; stop traversal at inner node still delivers usable result
- Approach is not limited to hand tracking



# Video



Hierarchical matching on short real dataset