1 Prior work

Our work makes use of the single-image hand pose prediction work by Xu et al. (2013)

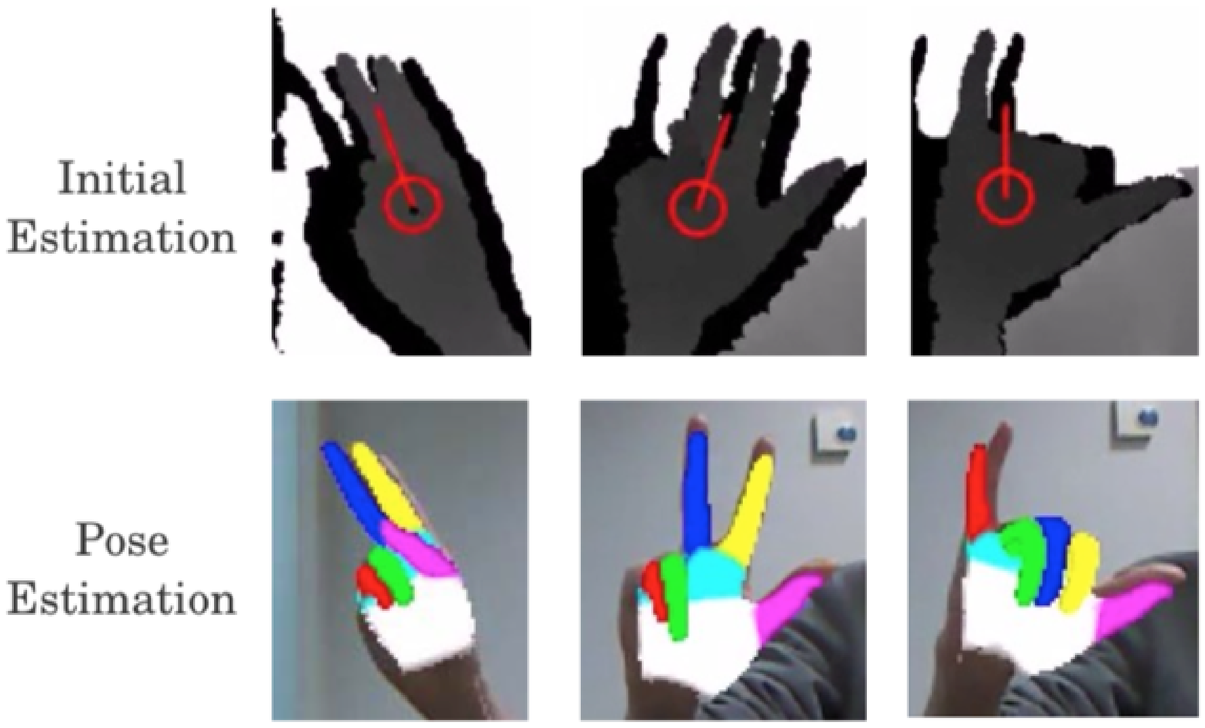


Figure 1

The hand pose estimation is done in three main steps:

1. A Hough forest (Gall et al, 2013), with a number of decision trees that combine input from a number of pixels to predict the lateral position of the base of the hand, as well as the orientation of the whole hand. This is used to determine the right rotational and lateral transformations to map the existing depth image to a canonicalized form of the depth image of the hand.
2. The rotational mapping produced in step 1 is applied over the per-pixel depth-invariant input features (Shotton et al, 2013) of a second Hough forest to create a number of 3D hand pose predictions.
3. The candidate predictions generated by step 2 are passed through an "estimation by synthesis" (Grenander, 2007) method that minimizes the distance between a synthetic image generated from the predicted model and the actual depth image provided.

2 Problem

This method works for some single-frame images. However, it often produces “noisy” results that are highly noticeable as “jitter” when using this method on a series of frames to produce a pose-estimation video. Since this method does single-frame prediction, it assumes complete independence between successive frames. In addition, the classifier occasionally creates a “flipped” prediction of the hand orientation. We present a few different methods to exploit this temporal correlation to produce smoother, more accurate pose estimation videos.

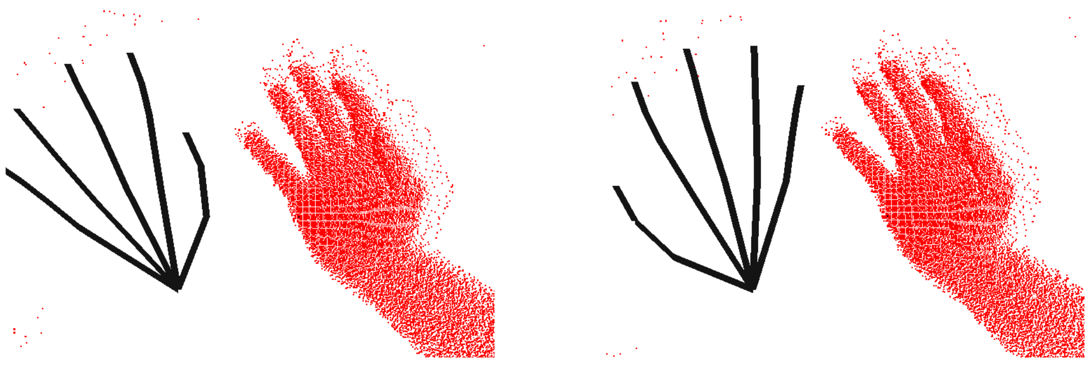


Figure 2: Black: the output of hand engine, red: the depth image. In the first frame the fingers are correctly classified, but in the next frame the fingers are mis-classified. This can be improved by exploiting the temporal correlation of two frames.

3 Algorithm

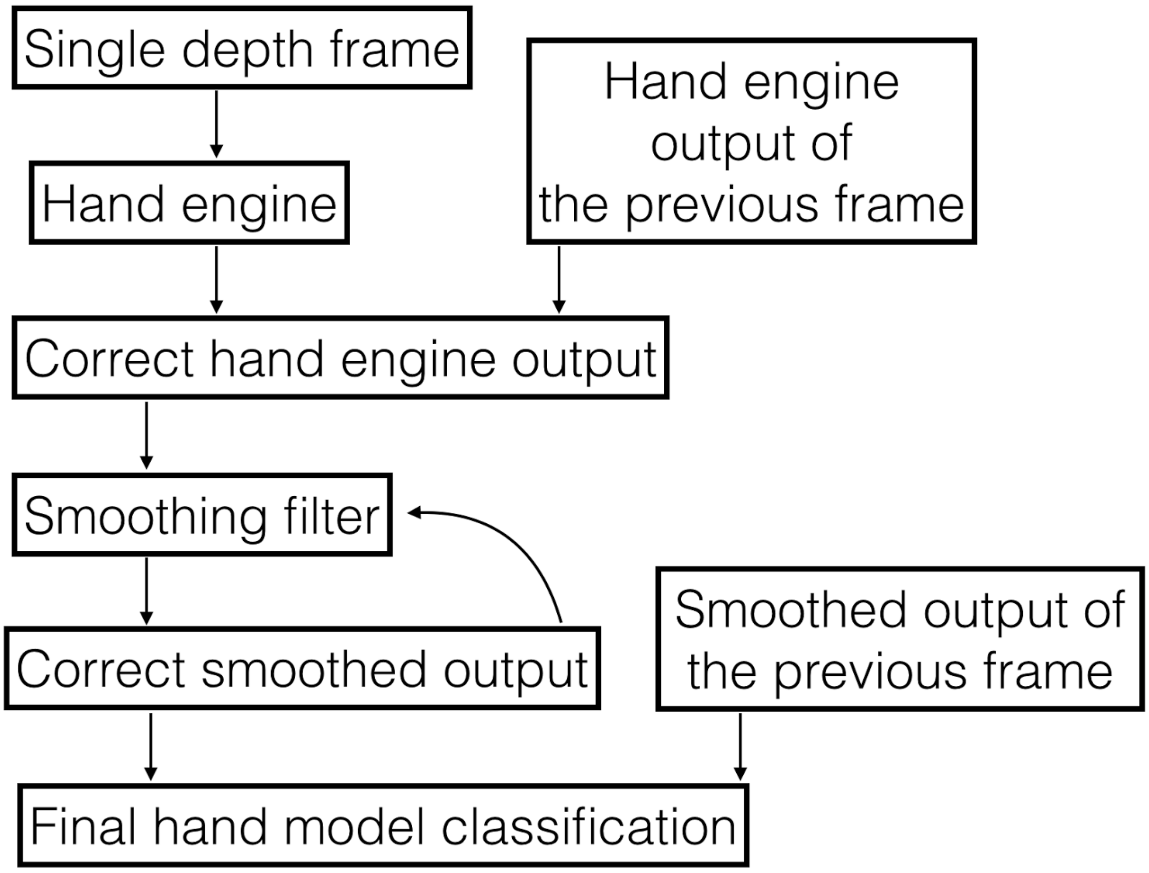

Single depth frame
Hand engine
Hand engine output of the previous frame
Correct hand engine output
Smoothing filter
Correct smoothed output
Smoothed output of the previous frame
Final hand model classification

Figure 3

- We build upon the "hand engine" from Xu et al (2013) by piping the output of it into another filter, consisting of a smoothing filter and a correction algorithm for fixing mis-classification of the fingers.
- Smoothing
  - Kalman filter (Kalman, 1960)
    - State: coordinates and velocity of every point
      - Velocity at frame i is assumed to be the displacement from the previous frame to the current frame
    - Transition: add the velocity vector to the coordinates vector
    - Problem: over-smoothing - "clamping" action does not appear to clamp fully after smoothing

  - · Solution: use angular velocity for all joints except the base of the hand.
  - · Angular velocity is represented as the rotation matrix that transforms the vector representing the bone in the previous frame to the one in the current frame.
  - · To make angular velocity-based transition linear, apply the angular velocity on the current frame to get a prediction for the next frame, and use the displacement from the current frame to the predicted frame as the velocity.

- Custom motion smoothing filter
  - Inspired by closed-loop proportional-derivative controllers (Minorsky, 1922).
  - Using first-order frame distance (velocity) and second-order frame distance (acceleration), the filter produces a smoothed output frame, trying to resist excessive acceleration, using the equation:
    - ∗ $x_t = x_t - 1 + (K_v \cdot v_t) + (K_a \cdot a_t)$
    - ∗ $v_t = z_t x_t - 1$
    - ∗ $a_t = v_t v_t - 1$
    - ∗ where z_t is the observed point cloud for time t, x_t is the "smoothed" output frame for time t, v_t is the first-order frame-displacement at time t and a_t is the second-order frame-displacement at time t. K_v and K_a are constants.
  - The parameters K_v and K_a are determined manually for now.
  - Problem: The filter produces jitter-free output, but in the case of flipped hand, the smoothed version results in a "squeezed" hand.

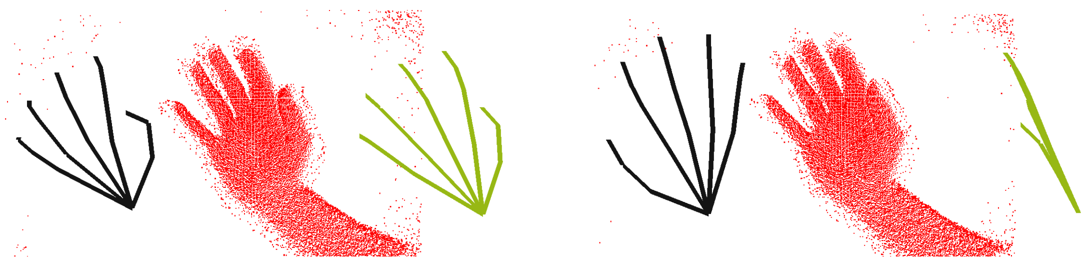


Figure 4: Black: the hand model predicted by the hand engine. Red: the depth image. Green: the smoothed hand model. Due to the misclassification of fingers, the smoothing algorithm attempts to even out the joint positions in between frames, resulting in the squeezed hand.

- Fix mis-classification of fingers
  - Assumption: metacarpophalangeal joints cannot be too close
    - Compute the maximum pairwise distance between these joints
    - If the current maximum distance is smaller than half of the maximum distance observed, consider the model invalid
  - Correct the input to the smoothing filter: get the mid-points between the joints in the previous frame and the joints in the current frame. If the hand model so constructed is invalid (by the definition given above), discard the current frame and use the previous frame as input to the smoothing filter.
  - If the output of the smoothing filter is invalid, discard it and use the smoothed output for the previous frame instead

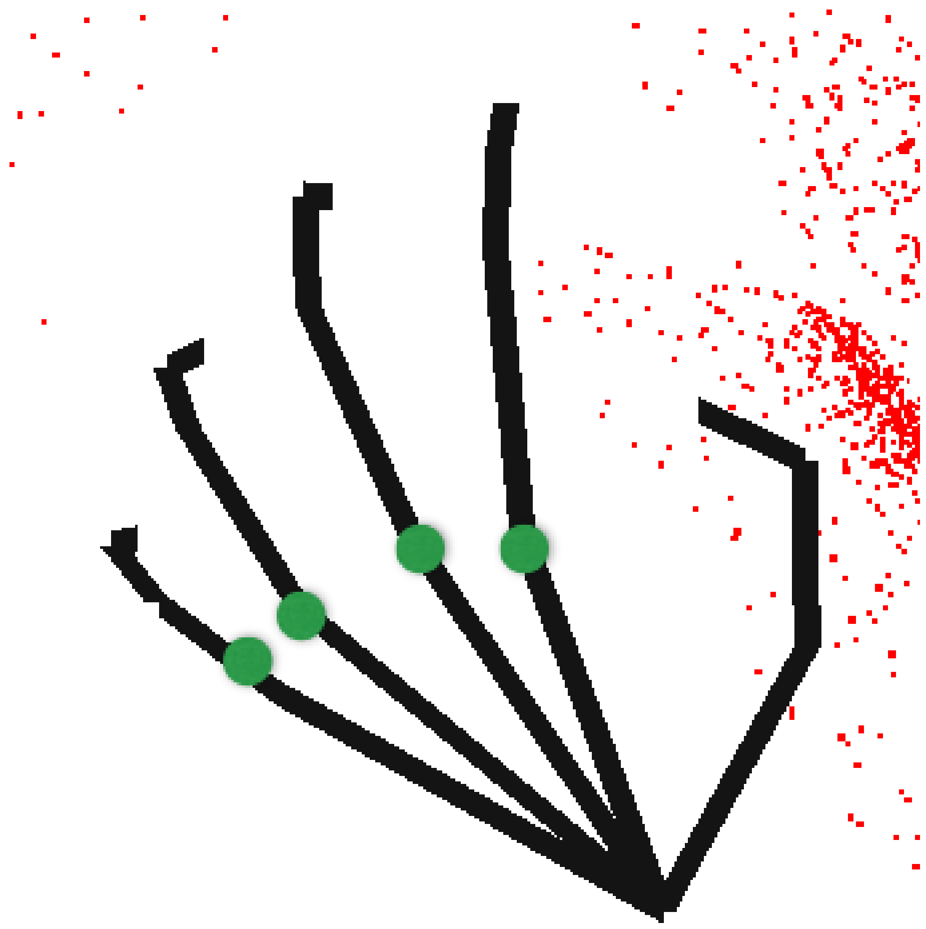
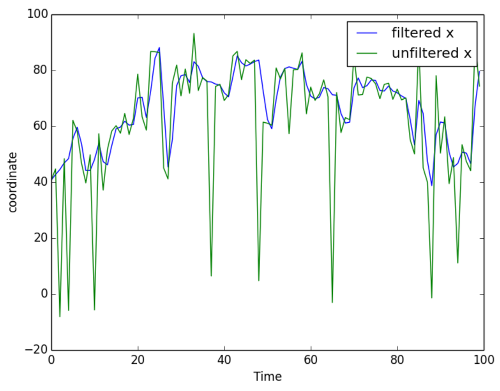
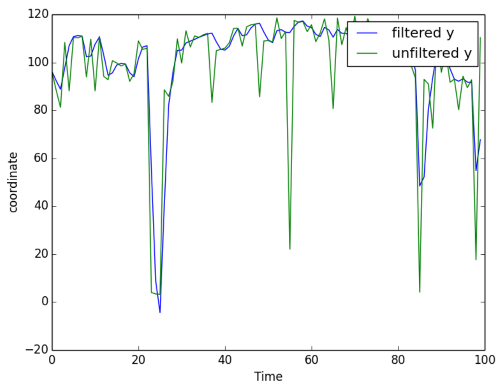


Figure 5: metacarpophalangeal joints (highlighted in green) are usually far apart, and it is assumed impossible for a human to squeeze them together

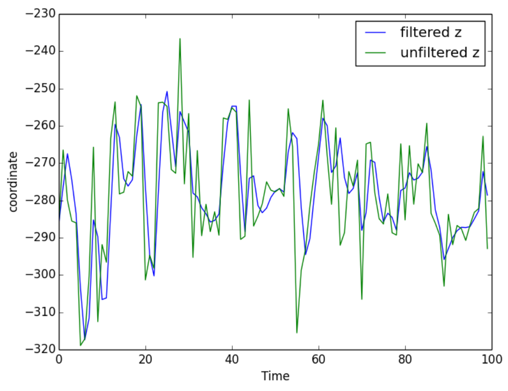
4 Illustration of smoothing



(a) x coordinate of the tip of the index finger



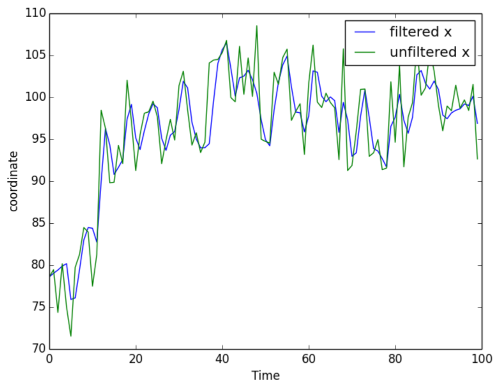
(b) y coordinate of the tip of the index finger



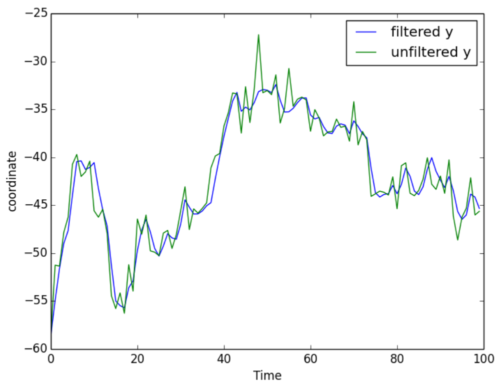
(c) z coordinate of the tip of the index finger

Figure 6: Coordinates of the tip of the index finger over time

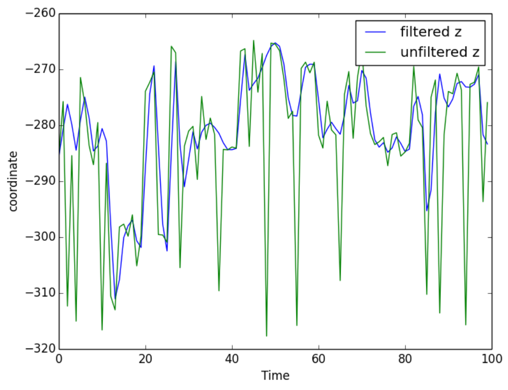
As seen, the trajectory of the fingertip is considerably smoother than the original output, and most instances of mis-classifications of fingers (manifested in sudden changes in position) are corrected.



(a) x coordinate of the base of hand



(b) y coordinate of the base of hand



(c) z coordinate of the base of hand

Figure 7: Coordinates of the base of the hand

5 Illustration of final output model

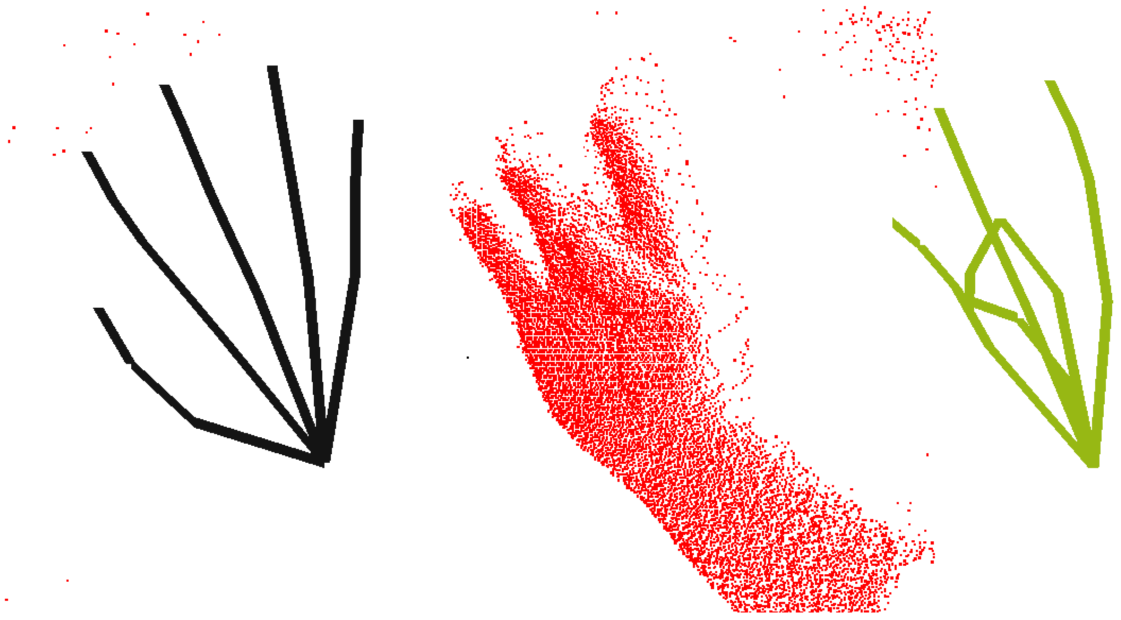


Figure 8: Red: depth image. Black: the classifier fails to classify the hand model, including the hand orientation (the thumb was identified to be on the left when in fact it is on the right). Green: by exploiting temporal information, we correctly identify the current hand gesture.

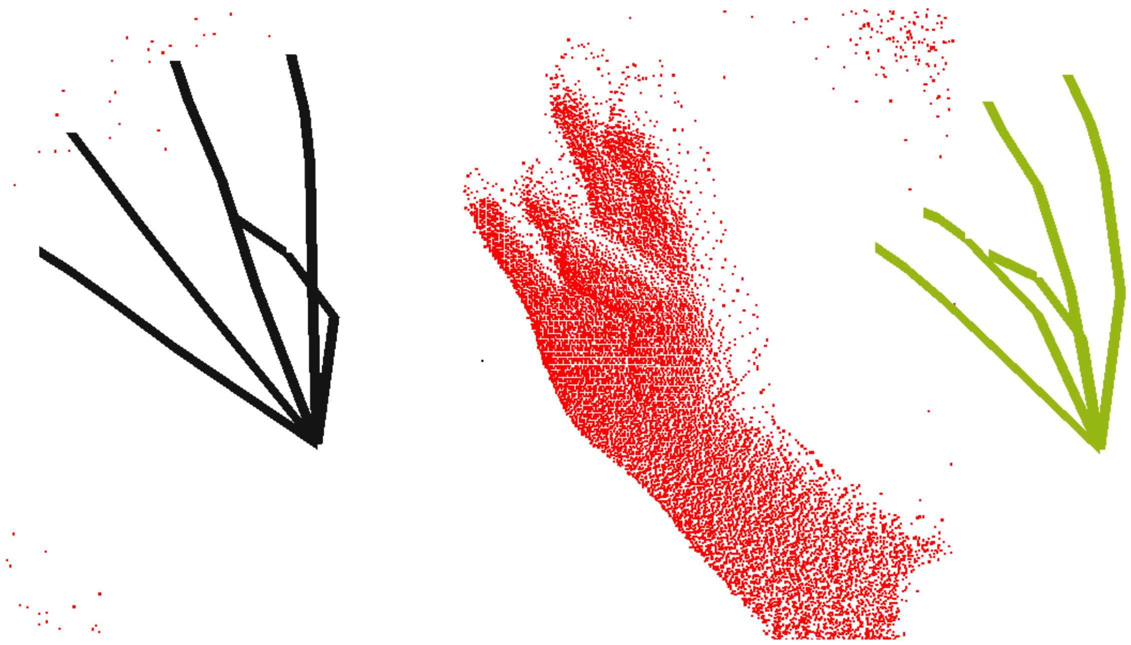


Figure 9: Red: depth image. Black: the classifier fails to classify the ring finger model. Green: by exploiting temporal information, we correctly identify the ring finger model.

6 Future works

The current method uses only one predicted outcome from the original hand engine, the result of "estimation by synthesis". The other predictions generated by the Hough forest are discarded in the process, leading to a less rich source of data for our algorithm.

We may instead be able to exploit the prediction candidates generated in step 2 of the hand engine to find the smoothest and most accurate joint trajectory.

7 References

Gall, Juergen, and Victor Lempitsky. "Class-specific hough forests for object detection." Decision Forests for Computer Vision and Medical Image Analysis. Springer London, 2013. 143-157.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. Journal of Fluids Engineering, 82(1), 35-45.

Minorsky., N. (1922), DIRECTIONAL STABILITY OF AUTOMATICALLY STEERED BODIES. Journal of the American Society for Naval Engineers, 34: 280–309. doi: 10.1111/j.1559-3584.1922.tb04958.x

Shotton, Jamie, et al. "Real-time human pose recognition in parts from single depth images." Communications of the ACM 56.1 (2013): 116-124.

U. Grenander. Pattern Theory: From Representation to Inference. Oxford University Press, 2007

Xu, Chi, and Li Cheng. "Efficient Hand Pose Estimation from a Single Depth Image." *Computer Vision (ICCV), 2013 IEEE International Conference on.* IEEE, 2013.