Learning Analysis-by-Synthesis for 6D Pose Estimation in RGB-D Images

Alexander Krull*, Eric Brachmann, Frank Michel, Michael Ying Yang, Stefan Gumhold, Carsten Rother

TU Dresden
Dresden, Germany
*alexander.krull@tu-dresden.de

1. Overview

The supplementary material is not necessary to understand the paper. This supplementary note discusses the following points:

- Further details on our training procedure.
- Detailed experimental results.
- Details on the calculation of the occlusion level.
- Additional images with qualitative results.

2. Further Details on our Training Procedure

We will now discuss the training procedure of our CNN. We use the *Torch 7* framework to implement our network. Our training procedure starts with a randomly initialized set of network parameters. We use the random initialization provided by *Torch 7*. To cover a greater range of possible energy values from the beginning, we multiply the weights in the last layer by 1000 before training starts.

During training we repeat the following steps:

1. Randomly pick five training samples from the training set.
2. Perform a gradient step for each of the training samples:
   (a) calculate partial derivatives $\frac{\partial}{\partial \theta_j} E\left(H_i, \mathbf{x}_i; \boldsymbol{\theta}^t\right)$ of the energy at the ground truth pose H_i using back propagation.
   (b) Run the inference scheme as described in Sec. 3.5 on the training image $\mathbf{x}_i$.
   (c) Use the result as initialization for Metropolis sampling described in Sec. 3.4.
   (d) calculate the partial derivatives $\frac{\partial}{\partial \theta_j} E\left(H_k, \mathbf{x}_i; \boldsymbol{\theta}^t\right)$ of the energy at each pose sample H_k and average the results.
   (e) Calculate the gradient of the log likelihood according to Eq. (4).
   (f) Calculate the new parameter set $\boldsymbol{\theta}^{t+1}$ using the gradient and current learning rate.
3. Use the current set of parameters $\boldsymbol{\theta}^t$ to perform inference on the validation set and determine the number of correctly estimated poses.
4. Update the learning rate similar to [1] as: $\gamma_t = 10^{-5}\gamma_0/(1 + \lambda t/5)$

In the end we pick the set of parameters which performed best on the validation set according to the number of correctly estimated poses. In the case where two parameter sets perform equally well, we pick the later one.

In our experiments we iterated the scheme described above 48 times and used the parameters $\gamma_0 = 10$ and $\lambda = 0.5$. For the proposal distribution in the Metropolis sampling scheme we used the covariance matrix $\Sigma_T = 25^{-1} I_3$ to sample of the translational component and the covariance matrix $\Sigma_R = 0.01^{-1} I_3$ to sample the rotational components.

In order to increase the number of training samples we randomly decide in each training step whether to rotate all images by 180deg.

3. Detailed Experimental Results

3.1. Occlusion Dataset by Hinterstoisser [3] and Brachmann [2]

Here we provide a table with detailed results on the *occlusion dataset* by Hinterstoisser [3] and Brachmann[2]. In Supplementary Table 1, we show the percentage of correctly estimated poses for the individual and training objects, as well as average values. The results correspond to Fig. 4 in the paper. The best results for each object are printed in bold numbers.

3.2. Dataset by Krull [4]

Here we provide a table with detailed results on the dataset by Krull *et al*. [4]. In Supplementary Table 2, we show the percentage of correctly estimated poses for the individual test and training objects, as well as average values. The results correspond to Fig. 6 in the paper. The best results for each object are printed in bold numbers.

4. Details on the Calculation of Occlusion

The performance of the evaluated methods depends strongly on how much of the object is visible in the im-

	Brachmann	Re-impl. of B.	Ours (*Tool_Box_1*)	Ours (*Cat_1*)	Ours (*Samurai_1*)	Ours (Average)
Ape	62.6%	64.1%	77.61%	77.09%	**79.06%**	77.92%
Can	80.2%	79.95%	84.26%	86.58%	**88.9%**	86.58%
Cat	50%	50.05%	54.78%	55.14%	**56.84%**	55.59%
Driller	84.3%	83.43%	**94.48%**	92.99%	93.32%	93.6%
Duck	67.6%	70.12%	71.1%	71.1%	**73.4%**	71.87%
Egg_Box	8.5%	11.17%	34.21%	**36.47%**	36.21%	35.63%
Glue	62.8%	**66.08%**	**72.59**	64.96%	66.08%	67.88%
Hole_P.	89.9%	90.33%	93.64%	95.37%	**95.29%**	94.77%
Average	63.24%	64.4%	72.83%	72.46%	**73.64%**	72.98%

Table 1. Detailed results on the *occlusion dataset* by Hinterstoisser [3] and Brachmann [2]. We provide the results of our method trained using three different objects/sequences (*Tool_Box_1, Cat_1, Samurai_1*) from [4].

	Brachmann *et al.*	Ours (*Tool_Box_1*)	Ours (*Cat_1*)	Ours (*Samurai_1*)	Ours (Average)
Cat_2	44.2%	58.28%	57.62%	**62.78%**	59.56%
Samurai_2	33.7%	51.99%	51.5%	**60.47%**	54.65%
Tool_Box_2	**59.4%**	57.69%	57.4%	52.96%	56.02%
Average	45.77%	55.99%	55.5%	**58.74%**	56.74%

Table 2. Detailed results on the dataset by Krull *et al.* [4]. We provide the results of our method trained using three different objects/sequences (*Tool_Box_1, Cat_1, Samurai_1*) from [4].

age and how much is occluded. To analyse this dependency, we calculated the percentage of occlusion for each object and image by rendering a depth image of the object in ground truth pose and doing a pixel wise comparison to the recorded depth image. We count a pixel as occluded whenever the rendered depth at the pixel is more than 50mm behind the recorded depth, or when there is no recorded depth value available for the pixel. To create Fig. 5 we divided the test images into bins according to their level of occlusion and calculated the percentage of correctly estimated poses for each bin. We used a bin width of 10%. In Supplementary Fig. 1 we use the same approach to show the average rotational error as a function of occlusion.

5. Additional Qualitative Results

Here we provide qualitative results for four test cases. Two cases are taken from each dataset. In Fig. 2 and Fig. 3 we show results for the *Can* and *Cat* object, respectively. They belong to the *occlusion dataset* of [3] and [2]. In Fig. 4 and Fig. 5 we show results for the *Samurai* and *Tool Box* object, respectively. They belong to the dataset from [4]. The upper four images in each figure show the RGB and depth channels as well as the forest predictions. The lower six are rendered and cropped images which are processed and fed into the CNN (see Fig. 2(e-g) in the paper) to calculate an energy vale. Object coordinates in all figures were mapped to the RGB cube for visualization.

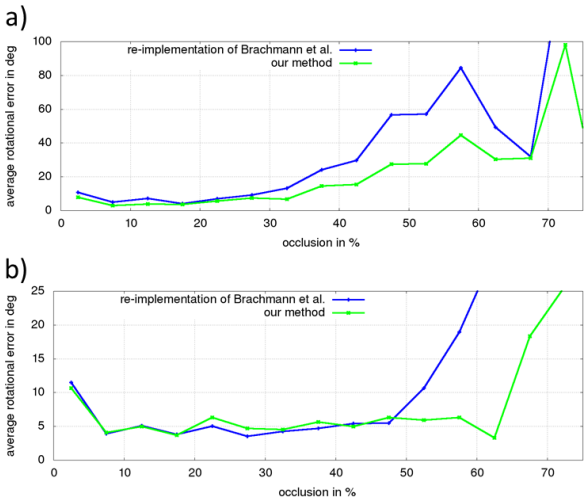

Figure 1. The average rotational error of pose estimates for the *Driller* object from the *occlusion dataset* of [3] and [2]. The CNN we used to achieve this results was trained with the *Samurai_1* sequence: We show the average error as a function of occlusion. The test cases are divided into bins with a width of 5%. (a) We consider all test cases for the *Driller* object. (b) We consider only test cases with correctly estimated pose according to the criterion from [3].

References

[1] L. Bottou. Stochastic gradient tricks. In *Neural Networks, Tricks of the Trade, Reloaded*, pages 430–445. 2012.

[2] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *ECCV*, pages 536–551, 2014.

[3] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *ACCV*, 2012.

[4] A. Krull, F. Michel, E. Brachmann, S. Gumhold, S. Ihrke, and C. Rother. 6-dof model based tracking via object coordinate regression. In *ACCV*, 2014.

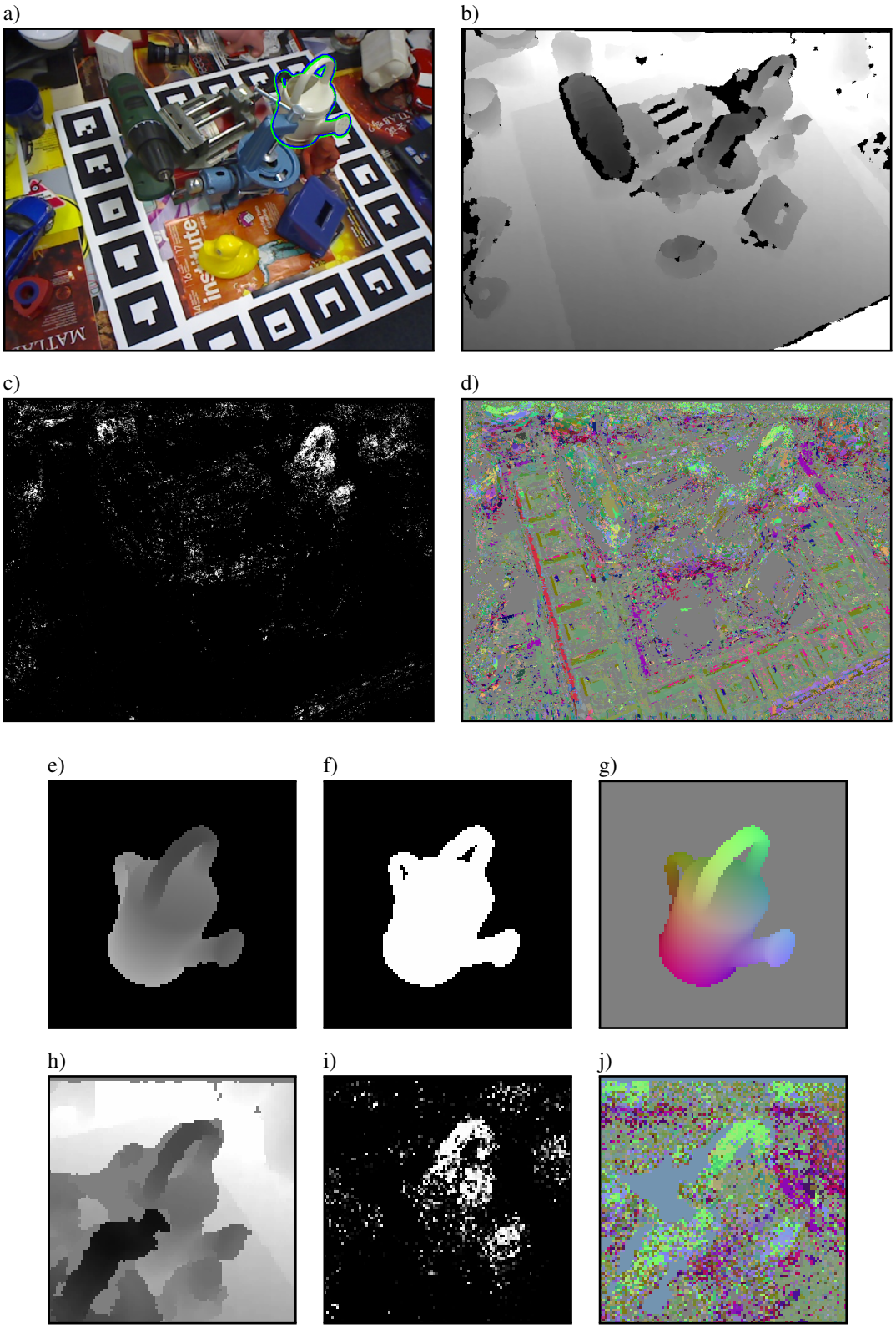

Figure 2. Qualitative results. (a) RGB image with results: blue indicates our pose estimate, green indicates the ground truth pose; (b) recorded depth image; (c) object probabilities predicted by the forest; (d) object coordinates predicted by one tree from the forest; (e) rendered depth image; (f) rendered mask; (g) rendered object coordinates; (h) cropped observed depth image (values which are more than the object diameter behind or in front of the object are shown as white or black respectively); (i) cropped object probabilities; (j) cropped predicted object coordinates.

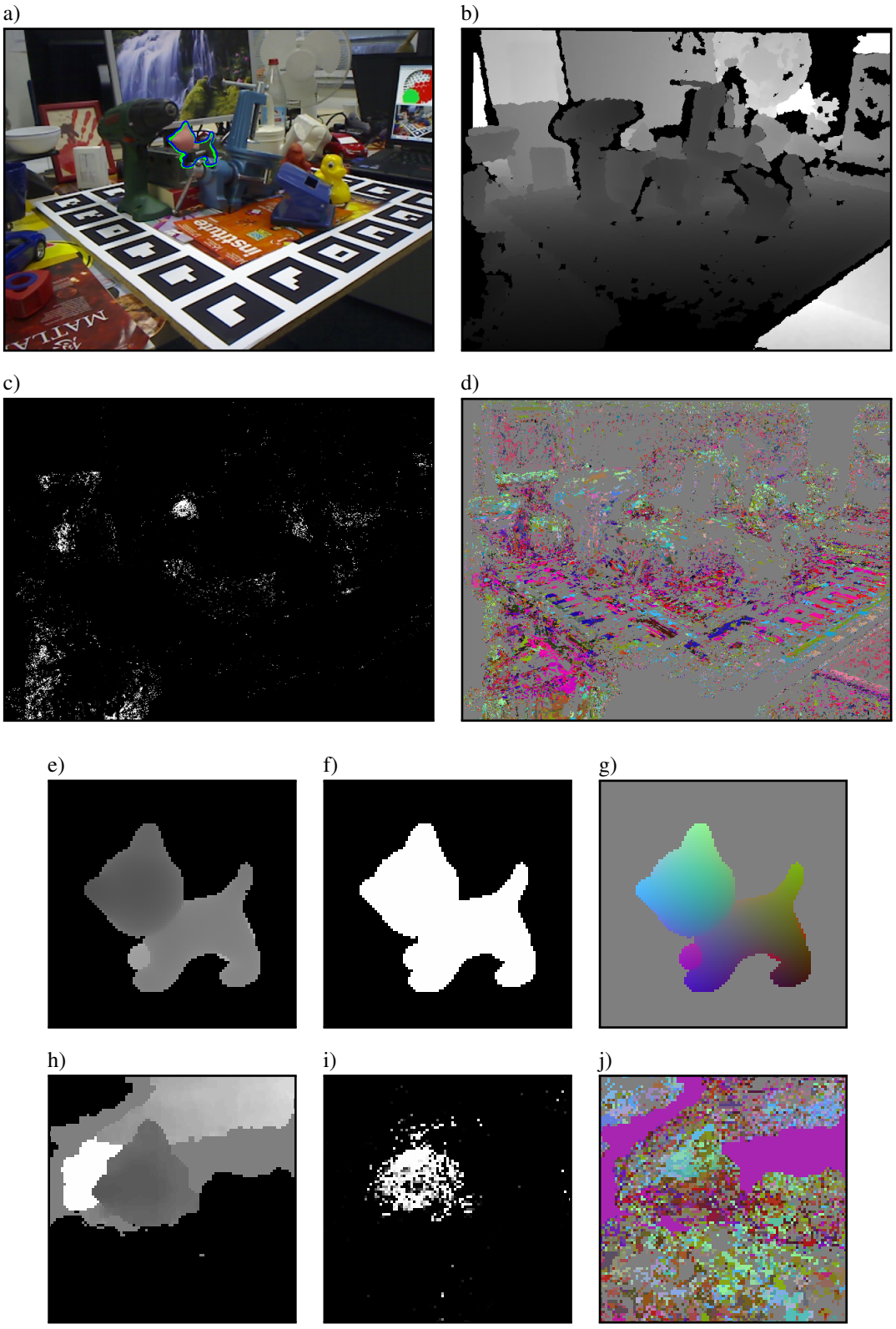

Figure 3. Qualitative results. (a) RGB image with results: blue indicates our pose estimate, green indicates the ground truth pose; (b) recorded depth image; (c) object probabilities predicted by the forest; (d) object coordinates predicted by one tree from the forest; (e) rendered depth image; (f) rendered mask; (g) rendered object coordinates; (h) cropped observed depth image (values which are more than the object diameter behind or in front of the object are shown as white or black respectively); (i) cropped object probabilities; (j) cropped predicted object coordinates.

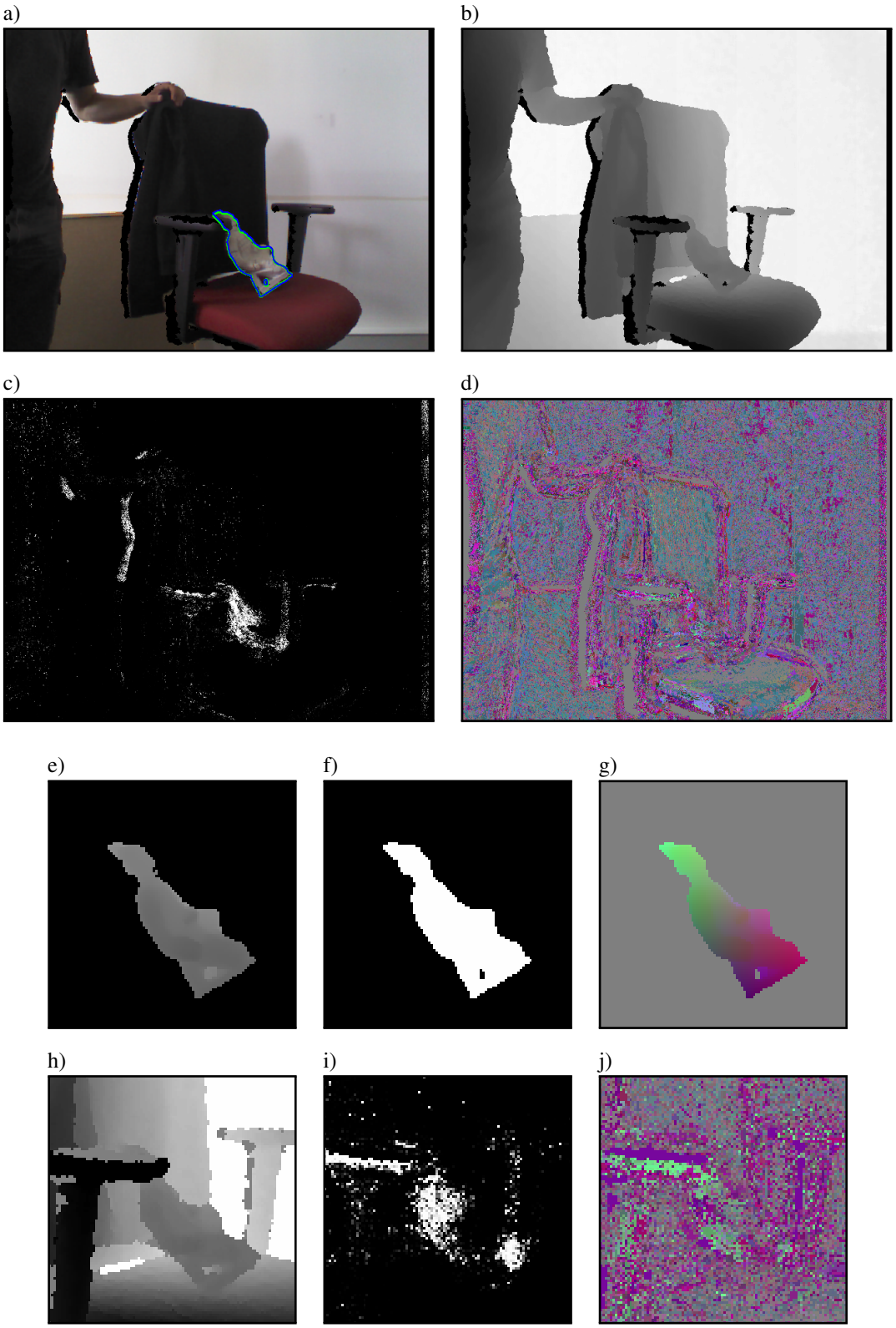

Figure 4. Qualitative results. (a) RGB image with results: blue indicates our pose estimate, green indicates the ground truth pose; (b) recorded depth image; (c) object probabilities predicted by the forest; (d) object coordinates predicted by one tree from the forest; (e) rendered depth image; (f) rendered mask; (g) rendered object coordinates; (h) cropped observed depth image (values which are more than the object diameter behind or in front of the object are shown as white or black respectively); (i) cropped object probabilities; (j) cropped predicted object coordinates.

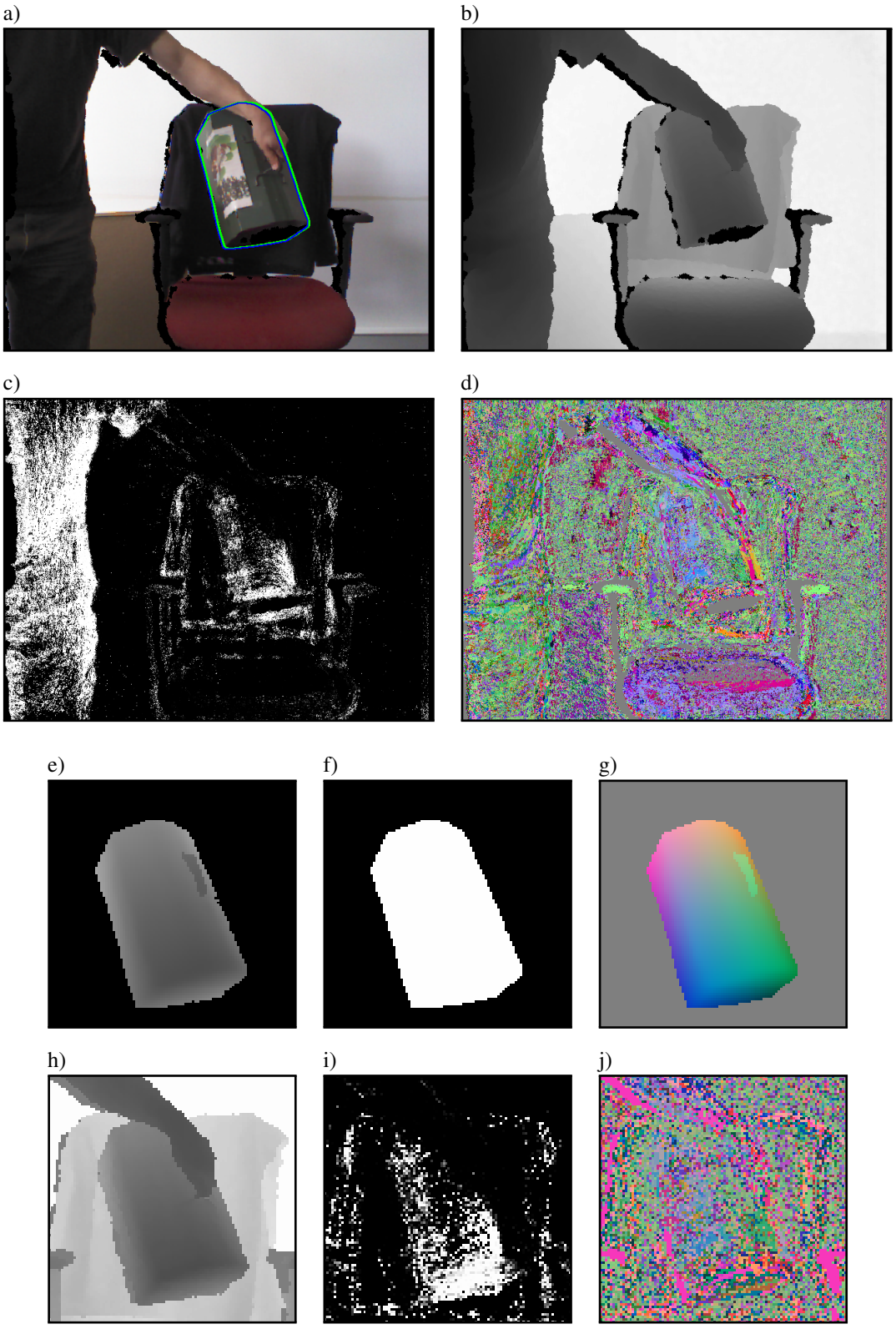

Figure 5. Qualitative results. (a) RGB image with results: blue indicates our pose estimate, green indicates the ground truth pose; (b) recorded depth image; (c) object probabilities predicted by the forest; (d) object coordinates predicted by one tree from the forest; (e) rendered depth image; (f) rendered mask; (g) rendered object coordinates; (h) cropped observed depth image (values which are more than the object diameter behind or in front of the object are shown as white or black respectively); (i) cropped object probabilities; (j) cropped predicted object coordinates.