Project Report

Computer Vision
Rigid Object Pose Estimation

Name: Devosmita Chatterjee
Personnummer: 910812-7748

Contents

1 Aim

The main aim of the project is to determine the pose estimation of each of the seven objects, relative to the camera. This is the resection problem.

2 Dataset

2.1 Images

9 images are provided.

2.2 9 .mat files

9 .mat files are provided each of which contains the following variables :
1. u – Cell array of 2D points for each object (normalized).
2. U – Cell array of 3D points for each object.
3. poses – Cell array of ground truth poses for each object.
4. bounding_boxes – Cell array of 3D corner points defining a bounding box for each object.

2.3 3 Functions

The following 3 evaluation or plotting (matlab) functions are provided :
1. minimalCameraPose.m
2. eval_pose_estimates.m
3. draw_bounding_boxes.m

3 Method

3.1 Description of the algorithm

We use the Direct Linear Transformation method (DLT) followed by the Levenberg-Marquardt (LM) method for estimating the pose.
Direct Linear Transformation method : Let X_i be the scene points and x_i be the respective projections. The aim is to solve the system of equations $\lambda_i x_i = PX_i$, i = 1,...,n where λ_i & $P = \begin{bmatrix} p_1^T \\ p_2^T \\ p_3^T \end{bmatrix}$ are the unknowns. Then we can write the system as

$X_i^T p_1 - \lambda_i x_i = 0$
$X_i^T p_2 - \lambda_i y_i = 0$. The system can be written in matrix form :
$X_i^T p_3 - \lambda_i = 0$

$$\begin{bmatrix} X_i^T & 0 & 0 & -x_i \\ 0 & X_i^T & 0 & -y_i \\ 0 & 0 & X_i^T & -1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \lambda_i \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} X_1^T & 0 & 0 & -x_1 & 0 & 0 & \dots \\ 0 & X_1^T & 0 & -y_1 & 0 & 0 & \dots \\ 0 & 0 & X_1^T & -1 & 0 & 0 & \dots \\ X_2^T & 0 & 0 & 0 & -x_2 & 0 & \dots \\ 0 & X_2^T & 0 & 0 & -y_2 & 0 & \dots \\ 0 & 0 & X_2^T & 0 & -1 & 0 & \dots \\ X_3^T & 0 & 0 & 0 & 0 & -x_3 & \dots \\ 0 & X_3^T & 0 & 0 & 0 & -y_3 & \dots \\ 0 & 0 & X_3^T & 0 & 0 & -1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

$$\Rightarrow Mv = \mathbf{0}$$

.
$min||r(v_0) + J(v_0)d||^2 + \lambda||d||^2$.
Levenberg-Marquardt method : We use the solution from DLT method as a starting solution and improve it using the Levenberg-Marquardt method. In the Levenberg-Marquardt method, we use the update rule $d = -(J(v_0)^T J(v_0) + \lambda I)^{-1}(J(v_0)^T r(v_0))$ where $\lambda = 0.1$.
The method of the project is briefly described in figure. 1.

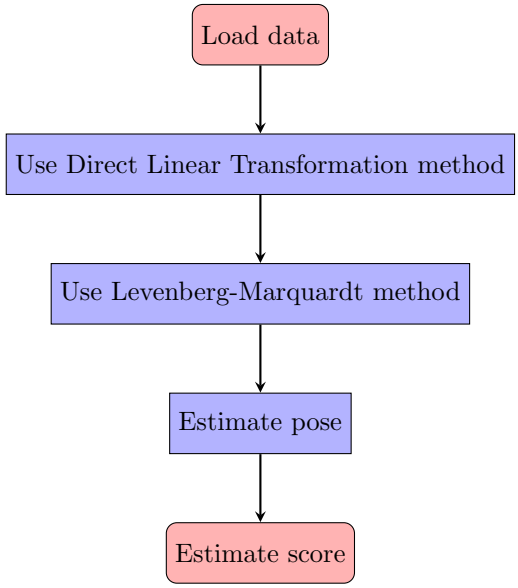

Figure. 1 Schematic overview of the method.

3.2 Corresponding scores for all objects in all images & total average of all scores in the table

Table 1: In this table, we present the corresponding scores for all objects in all images in pixels.

Total average of all scores = 4436

Objects					Scores				
	img1	img2	img3	img4	img5	img6	img7	img8	img9
ape	2131.79	8390.71	15127.26	2622.67	7433.64	12344.12	2436.35	11058.96	1971.33
can	944.04	826.57	963.18	7008.22	2030.37	1794.39	1968.04	2059.55	1257.18
cat	10728.70	2400.52	5230.35	2966.00	2313.67	591.28	789.76	2305.11	612.15
duck	3047.31	5167.73	2201.75	13979.26	8635.66	7953.20	1697.00	5010.28	2340.84
eggbox	1713.63	1951.78	1714.85	2897.67	1737.40	22442.10	6753.59	1814.57	868.35
glue	6818.49	14517.94	13239.40	6478.36	3000.64	4853.32	3824.73	3958.03	5219.34
holepuncher	5374.19	1282.20	5513.24	1245.73	224.75	442.25	367.78	472.99	488.95
Average scores	4394	4933	6284	5314	3625	7203	2548	3811	1814

3.3 Contribution to the implementation(s) of the algorithm(s) considered

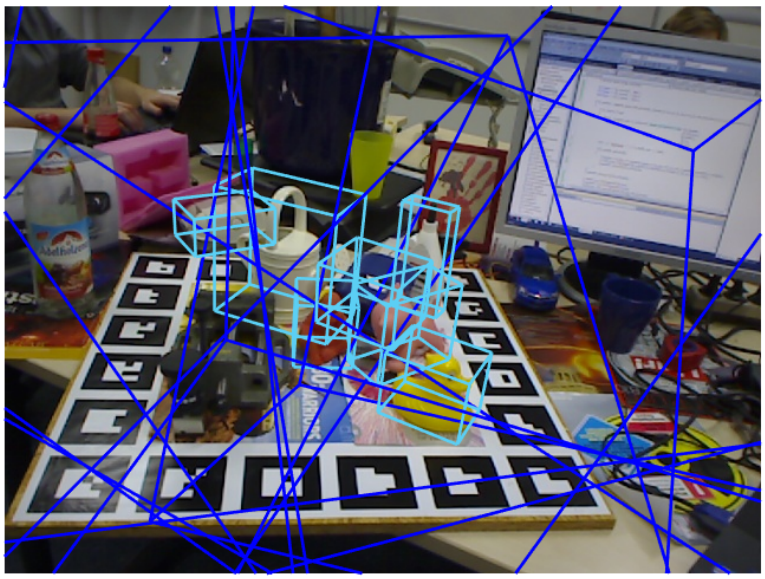

Figure. 2 Image1

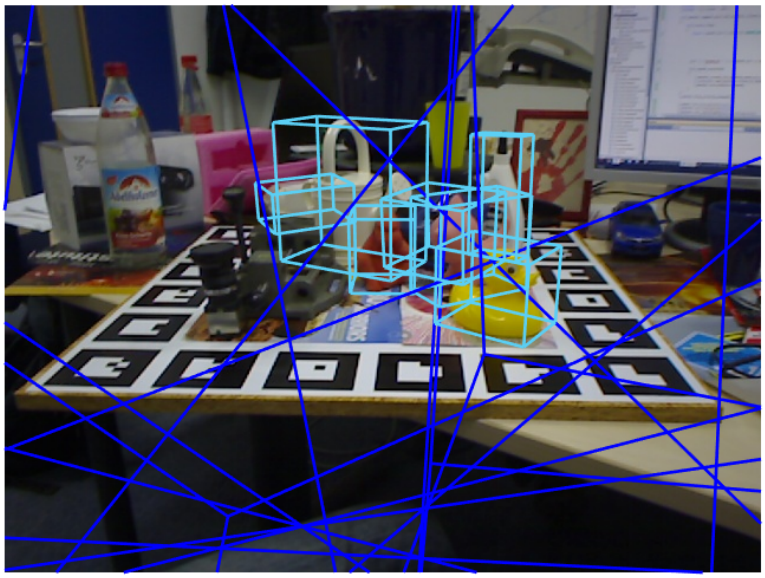

Figure. 3 Image2

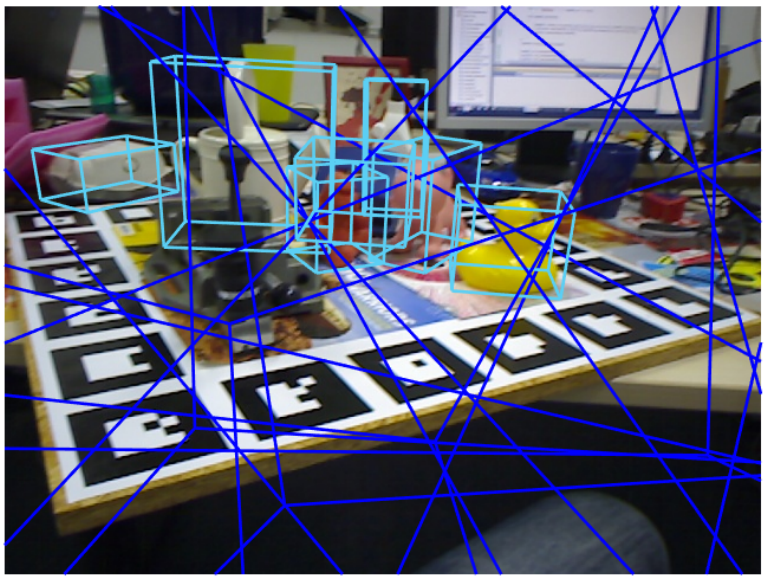

Figure. 4 Image3

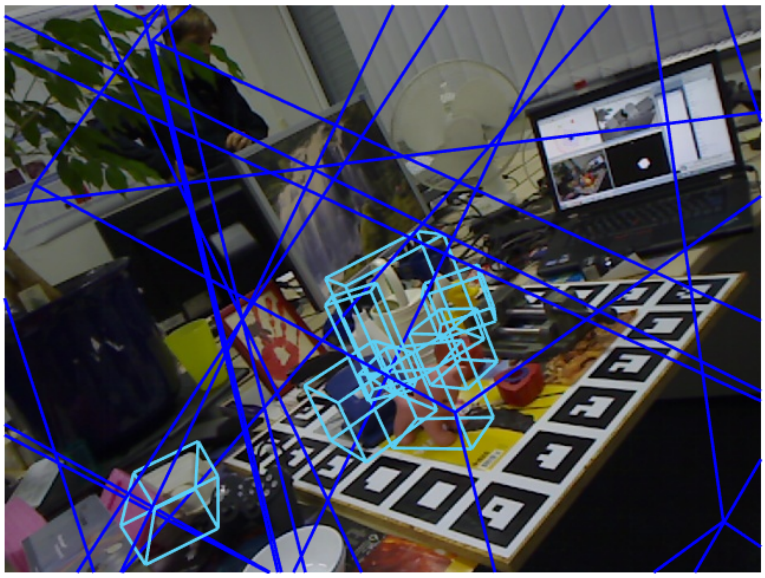

Figure. 5 Image4

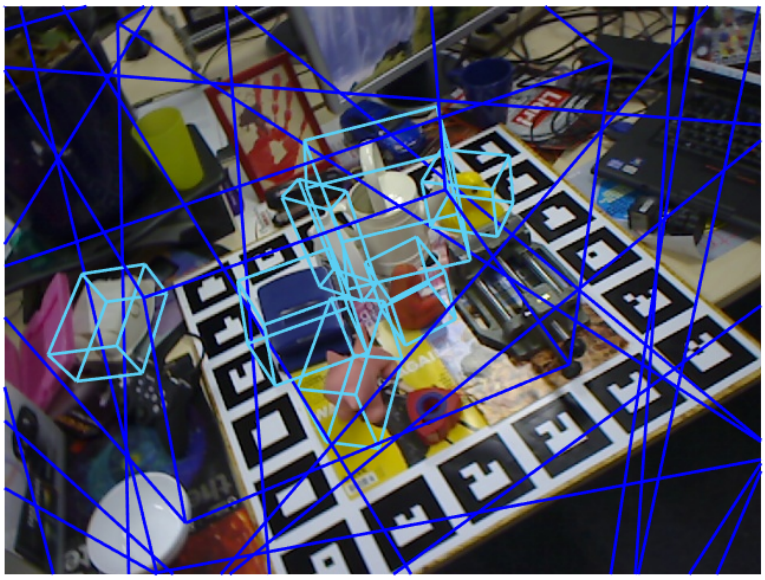

Figure. 6 Image5

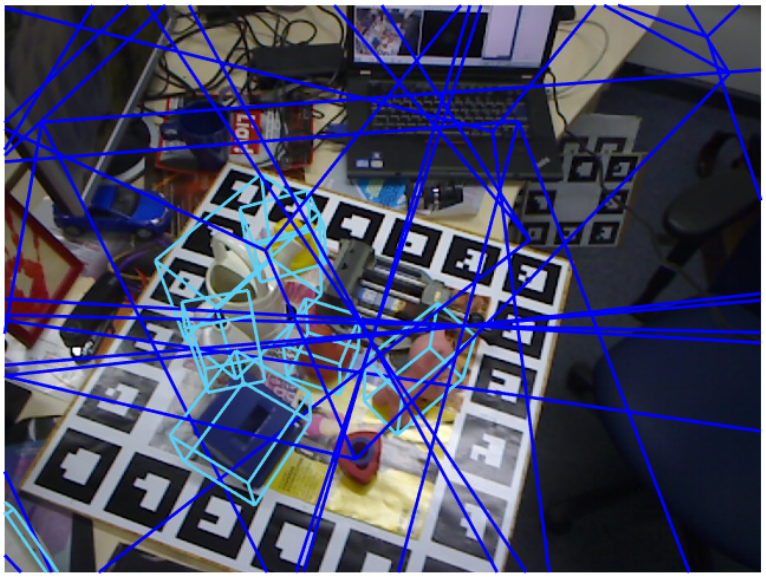

Figure. 7 Image6

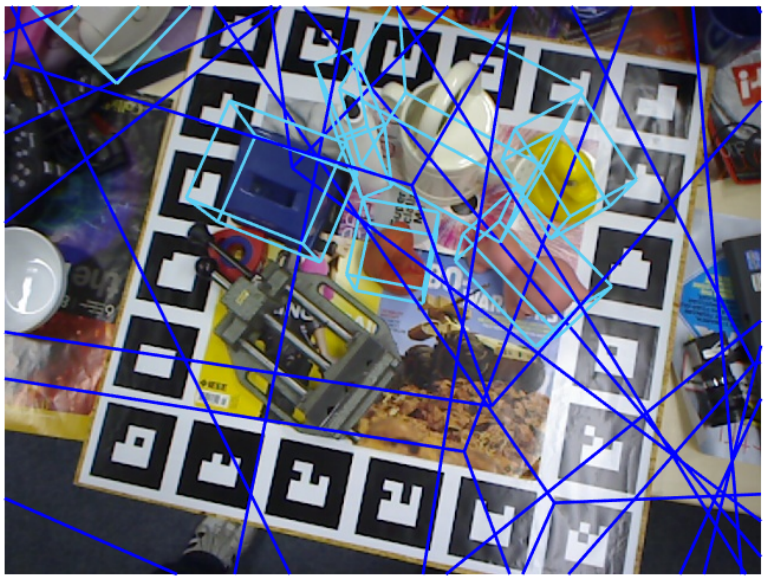

Figure. 8 Image7

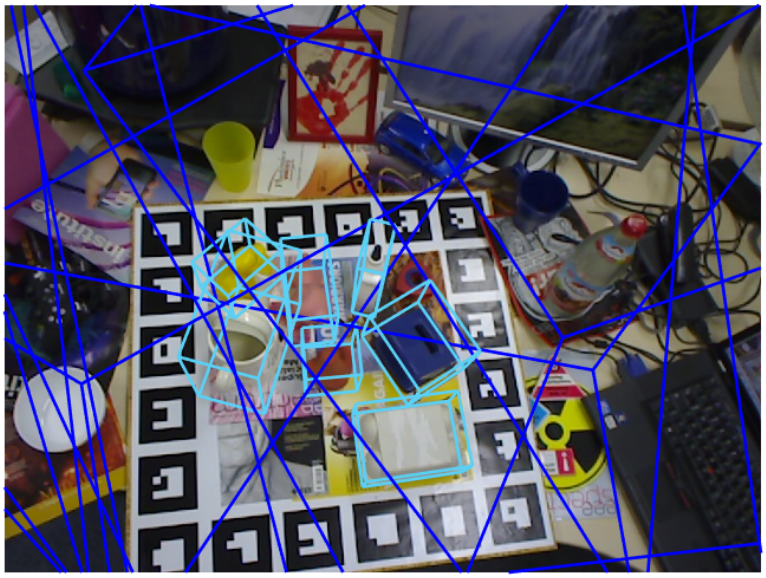

Figure. 9 Image8

Figure. 10 Image9

4 Matlab files

For image 1, the Matlab files are compEx1a.m, compEx1b.m and compEx1c.m.
For image 2, the Matlab files are compEx2a.m, compEx2b.m and compEx2c.m.
For image 3, the Matlab files are compEx3a.m, compEx3b.m and compEx3c.m.
For image 4, the Matlab files are compEx4a.m, compEx4b.m and compEx4c.m.
For image 5, the Matlab files are compEx5a.m, compEx5b.m and compEx5c.m.
For image 6, the Matlab files are compEx6a.m, compEx6b.m and compEx6c.m.
For image 7, the Matlab files are compEx7a.m, compEx7b.m and compEx7c.m.
For image 8, the Matlab files are compEx8a.m, compEx8b.m and compEx8c.m.
For image 9, the Matlab files are compEx9a.m, compEx9b.m and compEx9c.m.