Computer Vision
Assignment# 3

Submitted by
Shubham Kumar(21-944-491)

18 November, 2021

Contents

List of Figures

1 Calibration

1.1 *Implementation Details*

Q. Give a brief explanation of the approach in your own words in the report:

We are given 2D - 3D correspondences. As a first step these points are normalized and corresponding transformation matrices are stored for de-normalization later. Using these correspondences a constraint matrix is formed.

Derivation of constraint matrix: Let $\boldsymbol{x_i}$ be the image point corresponding to the world (3D) point $\boldsymbol{X_i}$, and $\boldsymbol{P}$ be the projection matrix. We know that $\boldsymbol{x_i} = \boldsymbol{P}\boldsymbol{X_i}$ (up to scale) should be satisfied. This constraint can be represented as cross product being zero (parallel vectors) *i.e.* $\boldsymbol{x_i} \times \boldsymbol{P}\boldsymbol{X_i} = 0$.

For, and $\boldsymbol{P} = \begin{bmatrix} P^{1T} \\ P^{2T} \\ P^{3T} \end{bmatrix}$, P^{iT} is the i^{th} row of $\boldsymbol{P}$.

Now, $\boldsymbol{x_i} \times \boldsymbol{P}\boldsymbol{X_i} = \boldsymbol{x_i} \times \begin{pmatrix} P^{1T}\boldsymbol{X_i} \\ P^{2T}\boldsymbol{X_i} \\ P^{3T}\boldsymbol{X_i} \end{pmatrix} = \begin{pmatrix} y_i P^{3T}\boldsymbol{X_i} - w_i P^{2T}\boldsymbol{X_i} \\ w_i P^{1T}\boldsymbol{X_i} - x_i P^{3T}\boldsymbol{X_i} \\ x_i P^{2T}\boldsymbol{X_i} - y_i P^{1T}\boldsymbol{X_i} \end{pmatrix}$

This can be written in matrix form as: $\begin{bmatrix} \mathbf{0}^T & -w_i\boldsymbol{X_i}^T & y_i\boldsymbol{X_i}^T \\ w_i\boldsymbol{X_i}^T & \mathbf{0}^T & -x_i\boldsymbol{X_i}^T \\ y_i\boldsymbol{X_i}^T & x_i\boldsymbol{X_i}^T & \mathbf{0}^T \end{bmatrix} \begin{pmatrix} P^1 \\ P^2 \\ P^3 \end{pmatrix} = 0$

However, only two of the equations are linearly independent because the third row is obtained, up to scale, from the sum of x_i times the first row and y_i times the second. So we get,

$$\boldsymbol{\epsilon_i} = \begin{bmatrix} \mathbf{0}^T & -w_i\boldsymbol{X_i}^T & y_i\boldsymbol{X_i}^T \\ w_i\boldsymbol{X_i}^T & \mathbf{0}^T & -x_i\boldsymbol{X_i}^T \end{bmatrix} \begin{pmatrix} P^1 \\ P^2 \\ P^3 \end{pmatrix} = 0 \tag{1}$$

, which is the constraint represent in matrix form (for one 2D-3D correspondence). We want the error $\boldsymbol{\epsilon_i}$ to be ideally zero vector. For any number of such constraints the rows can be appended in the left matrix for each of the points.

This constraint matrix is decomposed (SVD) to find the initial values of the elements of the Projection Matrix (P_hat) , this is essentially the DLT step.

Using the intial values the P_hat is further refined by minimizing the re-projection error. After optimization is complete the obtained projection matrix is denormalized using the transformation matrices used earlier on points. This gives the final projection matrix P, which is decomposed into K, R, and C. It is ensured that the signs of elements of the K matrix are as expected and the polarity of R matrix is also as intended. C matrix is computed as nullspace of P. Using this the translation part t is computed as $-RC$.

1.2 *Inline Questions*

Q. Given the camera models that you saw in the lecture, which part of the full model is not calibrated with our approach?

Since the camera model for which we are finding the parameters considers the transformation from 3D world to image space to be linear, it does take into account the radial distortion (non linear effects). Hence non-linear part of the camera is not being modeled.

Q. Briefly explain the potential problems if we skip this (Data Normalization) step in our algorithm

If we skip data normalization then values of the vector elements may become very large compared to other vectors and elements and the last element which is 1 . Also, their products will quickly become large and thus their contribution to error measure will be very large which will reduce the signal corresponding to other vectors with lower values of the elements (data entries). Also, if the values are large the optimization would lean towards the trivial solution as the minima, which is not intended. Normalizing the vectors would fix these problems.

Q. How many independent constraints can you derive from a single 2D-3D correspondence?

We can derive two constraints from a single correspondence.

Q. How does the reported reprojection error change during opti-mization? Discuss the difference between algebraic and geometric error and explain the problem with the error measure e=x×PX in your report.

The reported reprojection error decreases as the optimization progresses (see- below). Since this is a geometric error measure, as the optimization progresses, the parameters are adjusted so as to bring the project points closer to the image points.

Algebraic error is a measure that may not be directly related to a geometric object or be statistically meaningful. e.g. for finding solution h for $Ah = 0$ using DLT, one minimizes $||Ah||$ but this does not directly relate to the points in image or space.

Geomoetric error on the other hand is directly linked to geometric objects, e.g. reprojection error , which is the actual distance between the projection and the corresponding image point.

The problem with $e = \epsilon = x \times PX$ is that it is an algebraic error, and we see that ϵ is dependent on w_i in above equation 1). As we know w_i related to the depth of the point X_i from the camera in the direction along the principal ray, which in turn directly relates to the focal length (f) of the camera. Therefore the presence of the focal length in the expression for algebraic error means that the DLT algorithm will be biased towards minimizing focal length at a cost of a slight increase in 3D geometric error, which is not desired.

```
Starting with scaled P_hat matrix:
Iteration: 1 Error: 0.0012431509757006047
Iteration: 2 Error: 0.0013388609227372589
Iteration: 3 Error: 0.000654776151494981
Iteration: 4 Error: 0.0008574042379501926
Iteration: 5 Error: 0.0007177018243928381
Iteration: 6 Error: 0.0006282812714472499
Iteration: 7 Error: 0.0006369719955575233
Iteration: 8 Error: 0.0006257765149792685
Iteration: 9 Error: 0.0006256209366016529
Iteration: 10 Error: 0.0006258949756354121
Iteration: 11 Error: 0.000625389102208695
Iteration: 12 Error: 0.0006253574278337467
Iteration: 13 Error: 0.000625367110880334
Iteration: 14 Error: 0.0006253538899291235
Iteration: 15 Error: 0.0006253538899291235
```

Q. Report your computedK,R, andtand discuss the reported re-projection errors before and after nonlinear optimization. Do your estimated values seem reasonable?

The errors and computed K, R etc. have been reported below. The estimated values seem reasonable as $det(R) = 1$ and $RR^T = I$, and $K[2,2] = 1$ (0 based index). Also the skew is almost 0 (after factoring in focal length), and the diagonal elements of K are positive.

```
Reprojection error before optimization (P_hat): 0.0006316
Reprojection error before optimization (P_hat when scaled so that P_hat[2,3]=1): 0.

Reprojection error after optimization: 0.0006253
 P_hat =
 [[ 2.605e-01 -2.208e-01 -4.327e-04  3.318e-02]
 [-1.639e-01 -1.961e-01  4.434e-01  6.135e-02]
 [ 1.055e-01  1.316e-01  9.276e-02 -7.613e-01]]

P_hat_opt =
[[-3.421e-01  2.899e-01  6.869e-04 -4.334e-02]
 [ 2.158e-01  2.577e-01 -5.819e-01 -8.068e-02]
 [-1.401e-01 -1.729e-01 -1.222e-01  1.000e+00]]

K=
[[2.713e+03 3.313e+00 1.481e+03]
 [0.000e+00 2.710e+03 9.654e+02]
 [0.000e+00 0.000e+00 1.000e+00]]
R =
[[-0.774  0.633 -0.007]
 [ 0.309  0.369 -0.877]
 [-0.552 -0.681 -0.481]]
t = [[0.047 0.054 3.441]]
```

2 Structure from Motion

2.1 *Implementation Details*

We are given 10 images and correspondences for each pairs. By selecting two of the images, Essential Matrix (E) is computed. I chose images at index 5 and 6, because the default choice given in the code resulted in numerically unstable R, t values for the last camera and resulted in malformed reconstruction. Using the selected pair of images, and their correspondences, the Essential matrix is derived by solving DLT with constraints defined by the correspondences.

Derivation of constraints for the Essential Matrix Computation:

We know that Essential Matrix E is defined for "calibrated camera case" and it maps a point ($\boldsymbol{x}$) in one image to corresponding epipolar line in the second image. And thus one may write this constraint as:

$$\boldsymbol{x}'^{T} E \boldsymbol{x} = 0 \tag{2}$$

, because $\boldsymbol{x}'$, the corresponding point in the other image lies on the line $E\boldsymbol{x}$.

$\boldsymbol{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$, $\boldsymbol{x}' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$, and $E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix}$

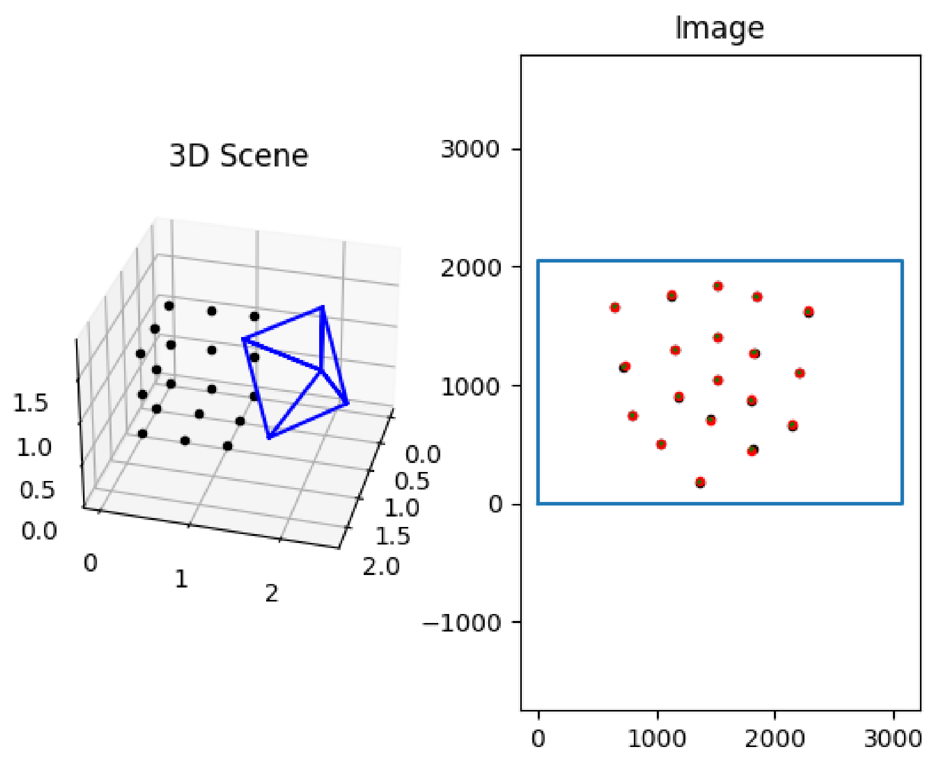

Figure 1: Calibration Results

$$\boldsymbol{x}'^T E\boldsymbol{x} = \begin{bmatrix} x' & y' & 1 \end{bmatrix} \begin{bmatrix} e_{11}x & e_{12}y & e_{13} \\ e_{21}x & e_{22}y & e_{23} \\ e_{31}x & e_{32}y & e_{33} \end{bmatrix}$$

Upon expansion it gives:

$$x'xe_{11} + x'ye_{12} + x'e_{13} + y'xe_{21} + y'ye_{22} + y'e_{23} + xe_{31} + ye_{32} + e_{33} = 0$$

$$(x'xe_{11}, x'ye_{12}, x'e_{13}, y'xe_{21}, y'ye_{22}, y'e_{23}, xe_{31}, ye_{32}, e_{33}) \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix} = 0$$

For n such correspondences we can write:

$$\begin{bmatrix} x_1^{'}x_1e_{11} & x_1^{'}y_1e_{12} & x_1^{'}e_{13} & y_1^{'}x_1e_{21} & y_1^{'}y_1e_{22} & y_1^{'}e_{23} & x_1e_{31} & y_1e_{32} & e_{33} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^{'}x_ne_{11} & x_n^{'}y_ne_{12} & x_n^{'}e_{13} & y_n^{'}x_ne_{21} & y_n^{'}y_ne_{22} & y_n^{'}e_{23} & x_ne_{31} & y_ne_{32} & e_{33} \end{bmatrix} \boldsymbol{e} = 0$$

where $\boldsymbol{e} = \begin{bmatrix} e_{11} & e_{12} & e_{13} & e_{21} & e_{22} & e_{23} & e_{31} & e_{32} & e_{33} \end{bmatrix}^T$

Thus the constraint matrix is created which can be solved using DLT to get the elements of E.

E is then decomposed to possible R and t values. This gives four possible configurations (adjusting for signs of R an t), out of which one is selected based on which configuration gives most points in front of the two cameras (using the *Triangulate-Points* function). (Note: one of the cameras (second camera) is assumed to have R=I and t=**0**)

The *TriangulatePoints* function perfroms DLT on constrains derived from key points of the two given images for which the 3d-points have not yet been registered to obtain 2D-3D correspondences. After preliminary solution, the points are again filtered based on whether they lie in front of the cameras or not.

Derivation of constraints for triangulation:

For projection of point $\boldsymbol{X}$ we have image point $\lambda\boldsymbol{x} = \boldsymbol{PX}$.

Let $\boldsymbol{P} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}$, P_i is the i^{th} row of $\boldsymbol{P}$

Now, $\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix} \boldsymbol{X}$ or, $\lambda = P_3\boldsymbol{X}$

$$\Rightarrow P_3\boldsymbol{X} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \boldsymbol{X}$$

$$\Rightarrow \begin{bmatrix} P_3x - P_1 \\ P_3y - P_2 \end{bmatrix} \boldsymbol{X} = \boldsymbol{0}$$

so for correspondence $\boldsymbol{x}^{'} \leftrightarrow \boldsymbol{x}$, we have

$$\Rightarrow \begin{bmatrix} P_3x - P_1 \\ P_3y - P_2 \\ P_3x' - P_1 \\ P_3y' - P_2 \end{bmatrix} \boldsymbol{X} = \mathbf{0}$$

which is the constraint matrix for finding the triangulation solution.

For the other images, Find2D3DCorrespondences is used to find intial 2D-3D correspondences using 3D-correspondences registered so far and 2D-2D correspondences between the current image and the images processed before it. Using the 2D-3D correspondence thus obtained, relative pose of the camera corresponding to the current image being processed is obtained using *EstimateImagePose.*

EstimateImagePose uses the 2D-3D correspondences and the K matrix to obtain R, t values for the camera.

Once R,t values of the new camera is obtained, further points are triangulated by using the correspondences of image points of current image with all other registered images so far. Using the newly triangulated points the resonstruction state is updated.

Find2D3DCorrespondences $\rightarrow$ *EstimateImagePose* $\rightarrow$ *UpdateReconstructionState* is repeated until all the images have been registered.

After this, the 3D points obtained thus far is plotted. We observe that in the 3D plot Fig[2], in addition to the main object, there is a bar like structure in the top right, which corresponds to some key points from 8th and the 9th images Fig[4].

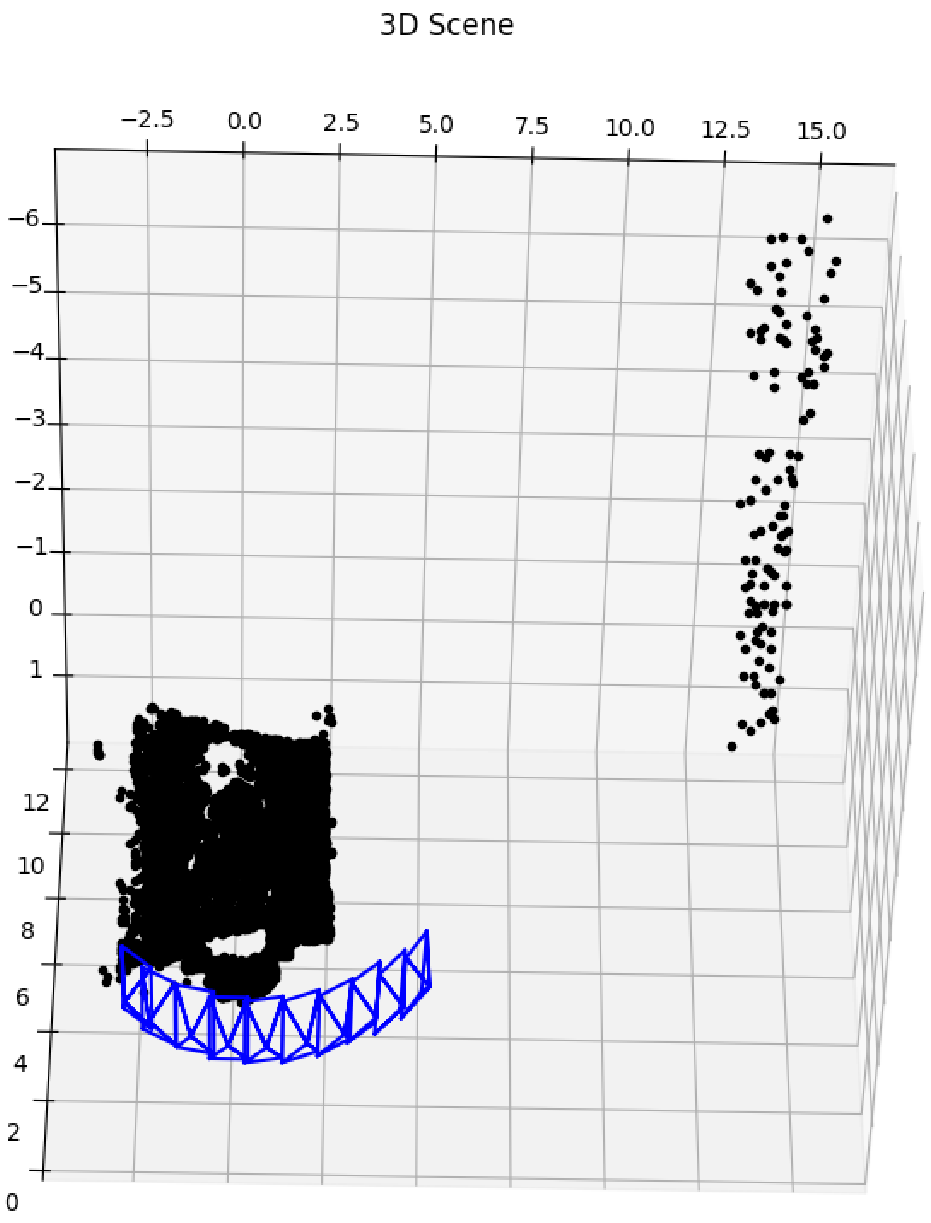

Figure 2: 3D Reconstruction

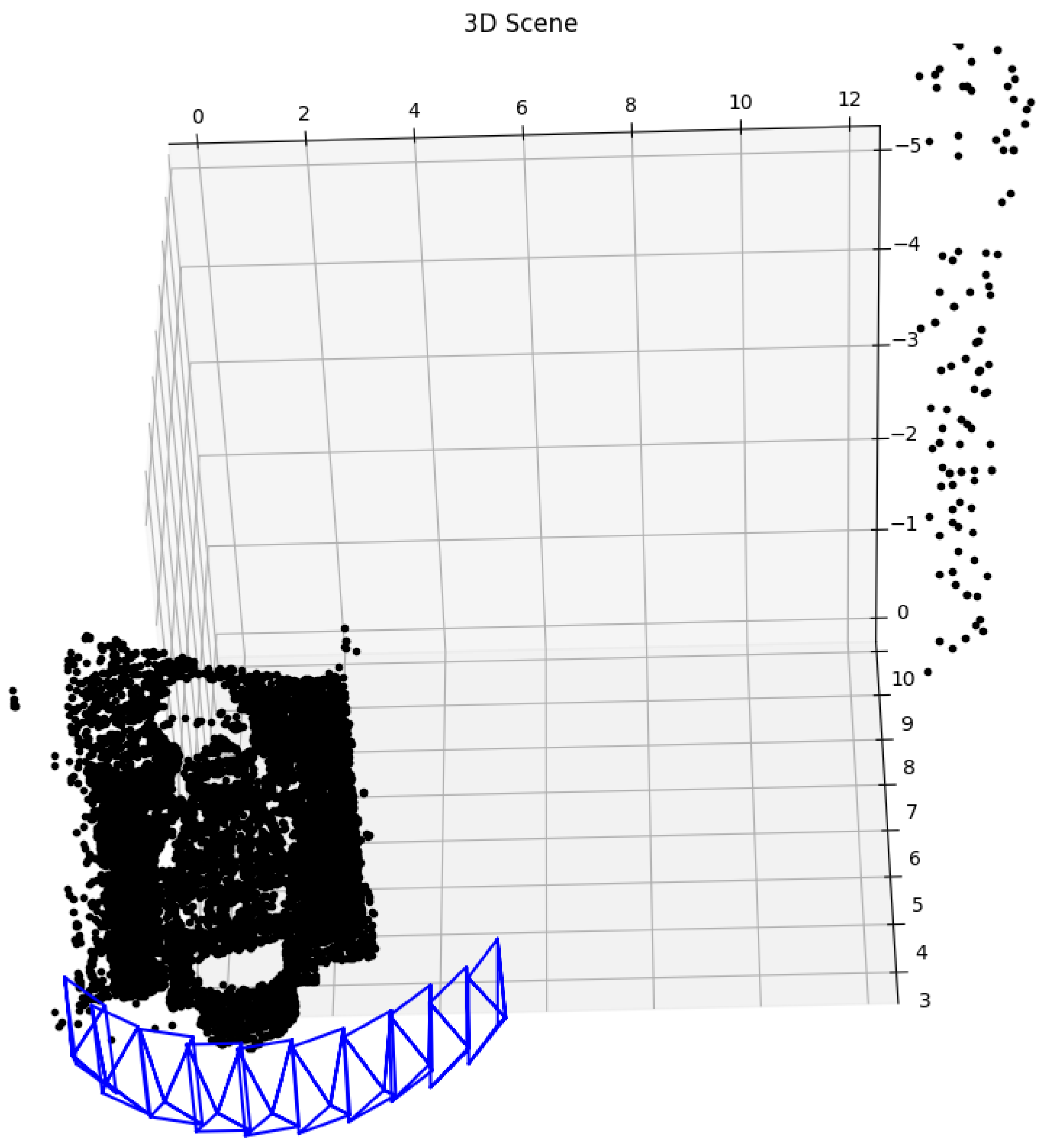

Figure 3: 3D Reconstruction (zoomed view)

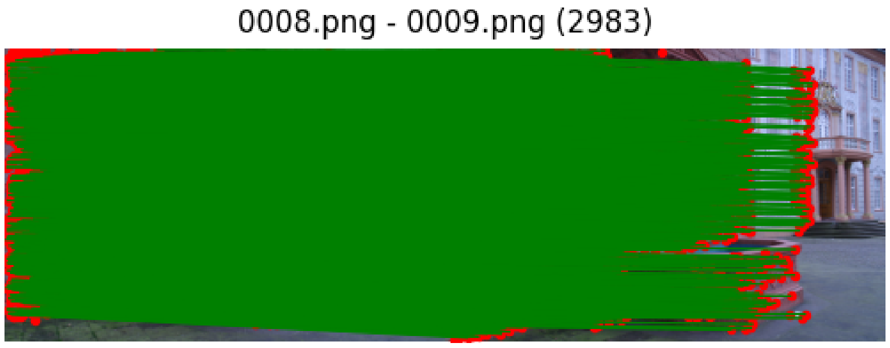

Figure 4: Correspondences for Image 8-9