Identifying Car Model from Photographs -

Fine-grained Classification using 3D Reconstruction and 3D Shape Registration

Xinheng Li
davidxli@stanford.edu

Abstract - *Fine-grained classification from photographs uses 2D image features, so this project explores the use of 3D reconstruction and 3D shape registration for identifying car model from images at arbitrary angles. 3D reconstruction for this project attempts 3D Euclidean reconstruction from correspondence points in images. 3D shape registration for this project attempts to match rigid reconstructed curves to curves in reference dataset which can differ by similarity transformation or being a subsection.*

1. Introduction

The problem of fine-grained classification, such as identifying the model of a car from its photographs, has been solved in many different ways. The ability to identify car model from image has many uses, such as captioning image or video, car statistics from Google Streetview or high resolution aerial photos, identifying cars in surveillance videos, or providing model detail to car hobbyists or potential buyer. However, most solutions are based on 2D image features, which constraints the 3D orientation of object relative to camera and other transformation allowed due to distortion and occlusion. For example, an existing mobile application requires the car to be photographed from relatively straight back side [1]. In fact, a sampling of entries in fine-grained classification competition, even for rigid and domain specific subject, still uses 2D image features, but offer very good performance. Also the use of 2D image features limits the training data to photographs or renderings of specific orientations, and the training result can be very large based on image size and feature density. This motivates the exploration of 3D reconstruction and 3D shape registration for fine-grained classification, which should overcome some of the limitations of the 2D feature based approaches, while posing input requirements and challenges of its own.

2. Previous Work

While there is no exact previous work of fine-grained classification using 3D reconstruction and 3D shape registration from 2D images, there are abundance of papers detailing each individual steps. Also there is a paper about augmenting 2D image feature with estimated 3D geometry.

2.1. Review of Previous Work

The 2D image patch feature rectified and augmented with 3D geometry for fine-grained classification of cars approach is detailed in [2]. It matches 2D projection of generic 3D models to keypoints in the image, and then uses the 3D geometry for image patch rectification. Then it uses RootSIFT for feature detection & description, represented along with 3D geometry using 3D spatial pyramid and BubbleBank, and finally uses SVM for classification.

3D reconstruction and rectification is an important of stage of this project and there are abundance of papers detailing various approaches. One approach seemed very interesting in that it promises 3D Euclidean reconstruction with uncalibrated cameras [3]. This approach starts being different by assuming perspective camera and uses iteration to find best fit of projective depth per camera image point. Then it uses this projective

depth and constraints on camera intrinsic and extrinsic matrices to find a least squares solutions to the inverse of projective ambiguity, resulting in Euclidean reconstruction that only differs from true object by similarity ambiguity.

2.2. Contributions from this Work

This work presents a fairly unique approach to the classical fine-grained classification problem using well studied components of computer vision. It includes an implementation of 3D reconstruction and rectification in Matlab without using libraries or complex builtin functions, and a simple solution to the 3D shape registration problem that handles similarity ambiguity.

3. Summary of Technical Solution

To limit the scope of this project, the input of this project is set of correspondence points between two or more images that represent corners of edge detected lines found in a car image. Given the correspondence points, 3D reconstruction and rectification is performed to obtain a 3D Euclidean space reconstruction. The reconstruction is matched to existing database using 3D shape registration. The result of 3D shape registration matching is input to SVM for computing the certainty of input belonging to each car model class. These major steps are detailed in the following subsections.

3.1. 3D Reconstruction and Rectification

This stage implements the method described in [3], which is reviewed in Section 2.2, with some minor modifications for simplification. There are *n* cameras, and *m* correspondence points for each camera, and subscripts *i=1...n* and *j=1...m* . The reconstruction phase uses direct factorization of the *3n-by-m* measured image point matrix W_s scaled by projective depth λ_{ij} for each image point in each camera, and is iterated upon to find best fit of λ_{ij}, which is an important input to the rectification step. The scaled image point matrix W_s is constructed from image points coordinates u_{ij} and v_{ij} as below:

$$Ws = \begin{bmatrix} \lambda_{11}\,[u_{11}\ v_{11}\ 1]^T, & \dots & , \lambda_{1m}[u_{1m}\ v_{1m}\ 1]^T \\ \dots & \dots & \dots \\ \lambda_{n1}\,[u_{n1}\ v_{n1}\ 1]^T, & \dots & , \lambda_{nm}[u_{nm}\ v_{nm}\ 1]^T \end{bmatrix} \quad (2)$$

λ_{ij} is projective depth per image point

$$Ws = \begin{bmatrix} P_1 \\ \dots \\ P_n \end{bmatrix} \begin{bmatrix} x_{1x}, \dots, x_{mx} \\ x_{1y}, \dots, x_{my} \\ x_{1z}, \dots, x_{mz} \\ x_{1w}, \dots, x_{mw} \end{bmatrix}, \quad (3)$$

P_i is 3x4 projection matrix for each camera

x_j is 1x4 homogeneous coordinate object point

The steps of the iterative factorization described by [3] are:

1. Initialize λ_{ij} to all 1.
2. Construct current scaled measurement matrix *Ws* using Equation (2).
3. Factorize Ws using singular value decomposition, by computing *[u, s, v] = svd(Ws)* and reduce to rank 4 by truncation of *u, s, v*. Set projective motion matrix $P \check{=} u$ and projective shape matrix $x \check{=} s * v$.
4. Compute updated value of $\lambda_{ij} = P_{iz}\check{} * x_j$ where $P_{iz}\check{}$ is the third row of each $P_i\check{}$.
5. Repeat from step 2 is updated value of λ_{ij} differs significantly from previous value.

The goal of this iterative reconstruction process is to estimate the projective depth λ_{ij} that best fits Equation (2) and it's improved by back projecting the current projective motion and shape matrices. Some traditional reconstruction factorization ignores this projective depth by making $P_i\check{}$ a *2x4* matrix since they are targeting affine rather than perspective projection.

The projective ambiguity is shown in Equation (4) below, where H is a 4x4 linear projective transformation and P is the true projective transformation of the cameras and x is the Euclidean space object points.

$$Ws = \check{P}\ \check{x} = \check{P}\ H\ H^{-1}\ \check{x} = P\ x \quad (4)$$

As shown in [3], the projective transformation H can be solved for several cases with some assumptions on camera intrinsic parameters through rectification (aka. normalization) steps. The three cases are: i. unknown focal lengths; ii. unknown focal lengths and constant principal point; iii. unknown focal lengths, principal points, and aspect ratios;. In reality, square pixels and centered camera principal point is common so only case i of unknown focal lengths needs to be implemented. Equation (5) below shows the relation between projective and true motion matrices using projective transformation H which is solved in rectification using case i assumptions of 0 camera principal point and 1 aspect ratio.

$$P_i = [\ M_i\ |\ T_i\ ] = \check{P}_i\ [\ A\ |\ B\ ] = \check{P}_i\ H\ ,\ \text{where A is 4x3, B is 4x1} \quad (5)$$

First step of rectification is to solve for B by setting up a homogeneous system using Equations (6) below and find least squares solution using SVD.

$$T_{ix} = \check{P}_{ix}\ B,\ T_{iy} = \check{P}_{iy}\ B\ ,\ T_{iz} = \check{P}_{iz}\ B, \quad (6)$$
$$T_{ix}/T_{iz} = \Sigma_j(\lambda_{ij}\ u_{ij})/\Sigma_j(\lambda_{ij})\ ,\ \ T_{iy}/T_{iz} = \Sigma_j(\lambda_{ij}\ v_{ij})/\Sigma_j(\lambda_{ij})$$

Second step of rectification is to solve for *A* by using constraints on M_i and assumptions on camera intrinsics. Equations (7) lists the camera intrinsics matrices K_i and extrinsics matrices R_i and t_i and how they relate to Equation (5). μ_i is the unknown scaling factor per camera, since the Euclidean space reconstruction is aligned and centered to first camera and assumes unary scaling for first camera, it differs from true Euclidean space by a similarity transformation of scale, rotation and translation.

$$P_i = \check{P}_i\ [\ A\ |\ B\ ] = \check{P}_i\ H = [\ M_i\ |\ T_i\ ] = \mu_i\ K_i\ [\ R_i\ |\ t_i\ ] \quad (7)$$

$$K_i = \begin{bmatrix} f_i & 0 & c_{0ix} \\ 0 & \alpha_i f_i & c_{0iy} \\ 0 & 0 & 1 \end{bmatrix},\ R_i = \begin{bmatrix} I_i^T \\ J_i^T \\ K_i^T \end{bmatrix},\ t_i = \begin{bmatrix} t_{ix} \\ t_{iy} \\ t_{iz} \end{bmatrix}$$

c_{0i} is camera i's principal point, I_i, J_i, K_i are its rotation axes, t_i is its translation vector, f_i is its focal length and α_i is its aspect ratio

The original authors proposed to solve $M_i = P_i \check{A}$ from Equation (7) by arranging it into:

$$MM^T = \check{P}_i A A^T \check{P}_i^T = \check{P}_i Q \check{P}_i^T\ ,\ Q = AA^T \quad (8)$$

So that *Q* is 4x4 symmetric matrix and only has 10 parameters to solve for using least squares, then factorize Q into *A*. The constraints on *Q* is derived from constraints on M_i. Equations (9) below lists the general constraints on partial camera matrix M_i given the assumptions on square aspect ratio and centered principal point. m_{ix} is first row of M_i and so on. They are derived based on orthonormal basis vectors of rotation matrix R_i and substitution of unary and null parameter assumptions.

$$|m_{ix}|^2 = |m_{iy}|^2 \quad (9)$$
$$m_{ix} \cdot m_{iy} = 0$$
$$m_{ix} \cdot m_{iz} = 0$$

$m_{iy} \cdot m_{iz} = 0$
$|m_{z1}|^2 = 1$, this is an added constraint assuming first camera has unary scaling

With *A* and *B* solved, the Euclidean reconstruction can be recovered by Equations (4, 5): $x = H^{-1} x \check{} = [A \mid B]^{-1} x \check{}$.

3.2. 3D Shape Registration

The Euclidean reconstruction from section 3.2 still contains a similarity transformation ambiguity to the true object since it assumes object is aligned and centered to first camera and first camera has uniform scaling. So in order to properly find matches of the same object from separate reconstructions, it must be able to compute the similarity transformation between the two. Second challenge from using reconstruction from correspondence points detection from images or photographs is that the detected correspondence points will differ based on image, due to factors such as occlusion and extreme lighting conditions. The corners and sharp inflections in curves should be fairly stable between images excluding occlusions, such as from blob detectors, but points along a curve or smoother inflections will generally vary based on detector and image. So shape registration must be able to handle the case where the same curve is given by two different list of points, or one curve is a strict subset of the other. The use of perpendicular distances between a point and line segment in Equation (21) solves the problem of two different sets of points for the same curve, since one set can be interpolated into the other set.

A naive shape registration algorithm using Levenberg-Marquardt algorithm to solve a minimization problem is proposed. The objective function's input is the scale, Euler rotation angle and translation parameters of the similarity transformation between two objects. An object is defined as list of curves, and each curve is an ordered list of points, which might be reversed. Output of the objective function is the accumulated and normalized error of curving matching result. Body of the objective performs the following pseudocode:

```
H = similarityMatrixFromParameters(input);
fv = zeros(count of test_curves,1);
test_curves = H * original_test_curves;
foreach c in test_curves:
  c0 and ce are its endpoints;
  foreach r in unclaimed reference_curves:
    find segments s in r for which c0 and ce are closest to, using triangle
inequalities of |s0-se|≤|s0-c0|+|se-c0| and |s0-se|≤|s0-ce|+|se-ce|;              (20)
    record the smallest distances and their corresponding segments s0,se and curve
r.
  end
  claim curve r in reference_curves with smallest distance to c0 and ce;
  extract the curve section of r between s0 and se as rc, reverse the order as
necessary;
  initialize segment s as first segment of rc;
  dist = 0;
  foreach point p between endpoints of c:
    if p is closer to s+1 than s, set s=s+1;
    dist += |(se-s0)x(s0-p)| / |se-s0|;                                          (21)
  end
  dist /= size(c)-2;
  fv(c) = dist;
end
```

Fv is the output of the objective function, and LMA tries to minimize the norm of Fv. While LMA can handle nonlinear system, it's sensitive to the initial parameters and prone to local minimums. By thought experiment and actual experiment in Section 4.2, the most sensitive and most nonlinear parameter of the objective function is the rotation angles, which can easily trap LMA in local minimums that's far from the actual solution. Since rotation angles are limited in range, it's worth the computation time to perform a rough sweep of all rotation angles as the initial inputs to LMA. In contrast, scaling and translation are smoother contributions to the error and LMA should be able to minimize the error contribution from scaling and translation. After LMA finds a solution, the Fv can be used as feature vector to the final stage of classification, since each entry of Fv represents accumulated and normalized distance between a test curve and a reference curve.

Another approach to shape registration is also proposed, inspired by histogram of oriented gradients image feature. This approach will be suitable if the scaling of similarity ambiguity is uniform, which will not affect the angles between points of a 3D curve. Of course, global rotation and translation do not affect the angles between points of a 3D curve at all. A histogram of angles between segments of a curve can be constructed, and normalized by the number of points and length of segments. The complete feature vector of an input reconstruction is the list of histograms.

3.3. Classification using 3D Shape Registration Results

This stage should be implemented using SVM as one against all classification. The 3D shape registration step is performed against each reference car model database, producing a list of errors for each curve in the test object. These errors should be remapped to the list of reference curves, forming a fixed length sparse feature vector of errors to match each curve of reference car model. The SVM will be trained using positive and negative examples for each reference model, so that it can learn the weight of each curve in reference model in identifying the reference model, and be able to distinguish and ignore common or similar curves for all car models, such as the wheel wells.

4. Experiments

Matlab is used to implement the 3D Reconstruction and Rectification and 3D Shape Registration stages described in Sections 3.2 and 3.3. No external libraries are used and only basic builtin functions are used, and Matlab's fsolve() function is used to find least squares solution using Levenberg-Marquardt algorithm.
The implementation is available at https://drive.google.com/folderview?id=0B95r5M2m-E0gdDcySWdNVkZKRG8

4.1. 3D Reconstruction and Rectification

To obtain quantitative test results of 3D reconstruction and rectification, synthetic 3D model of a sedan from [8] is used to provide the ground truth of reconstruction, and the model is perspectively projected to obtain images of multiple views, and correspondence points simultaneously.

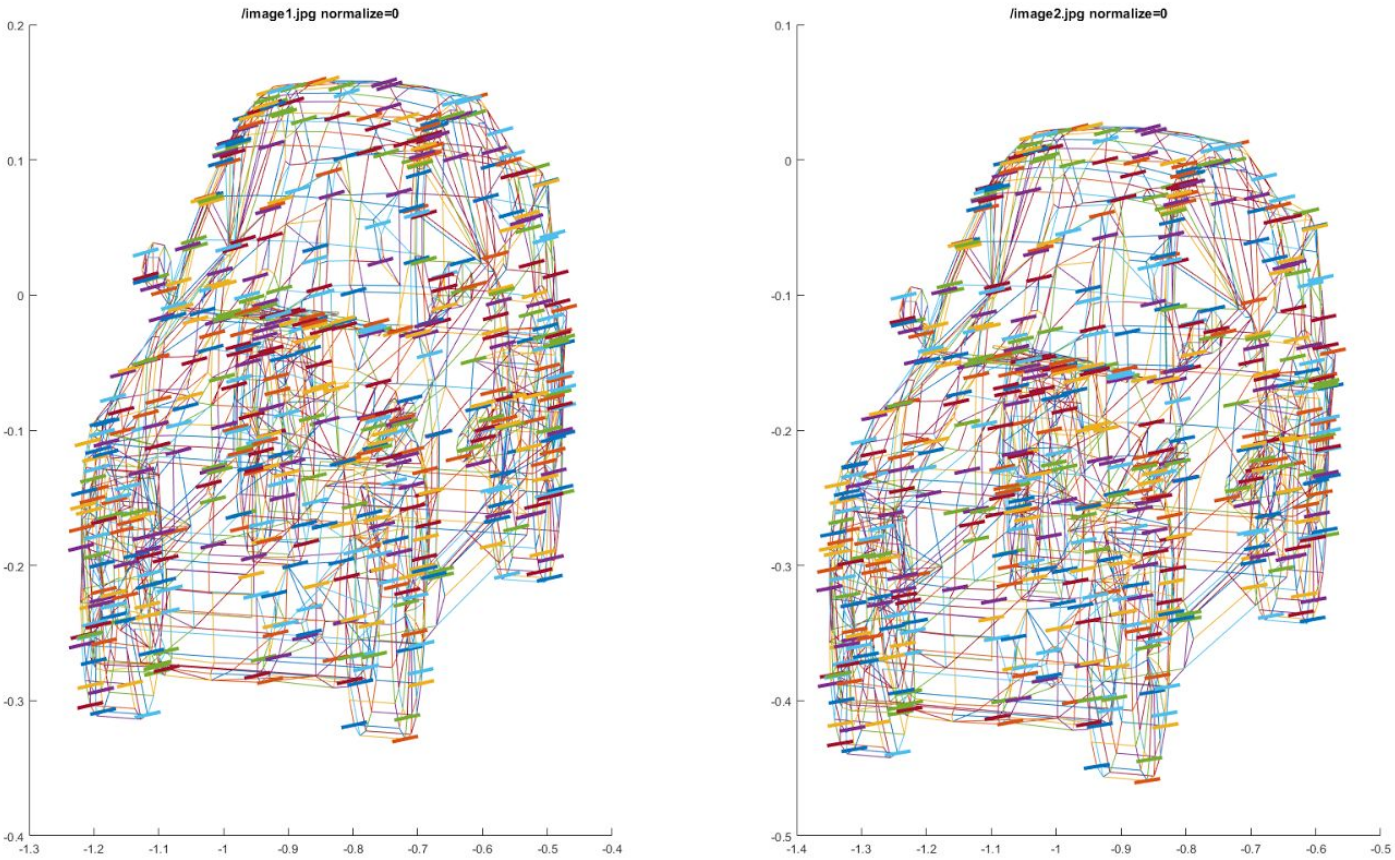

Fig. 2. Two views of wiremesh with correspondence points and epipolar lines highlighted by solid lines.

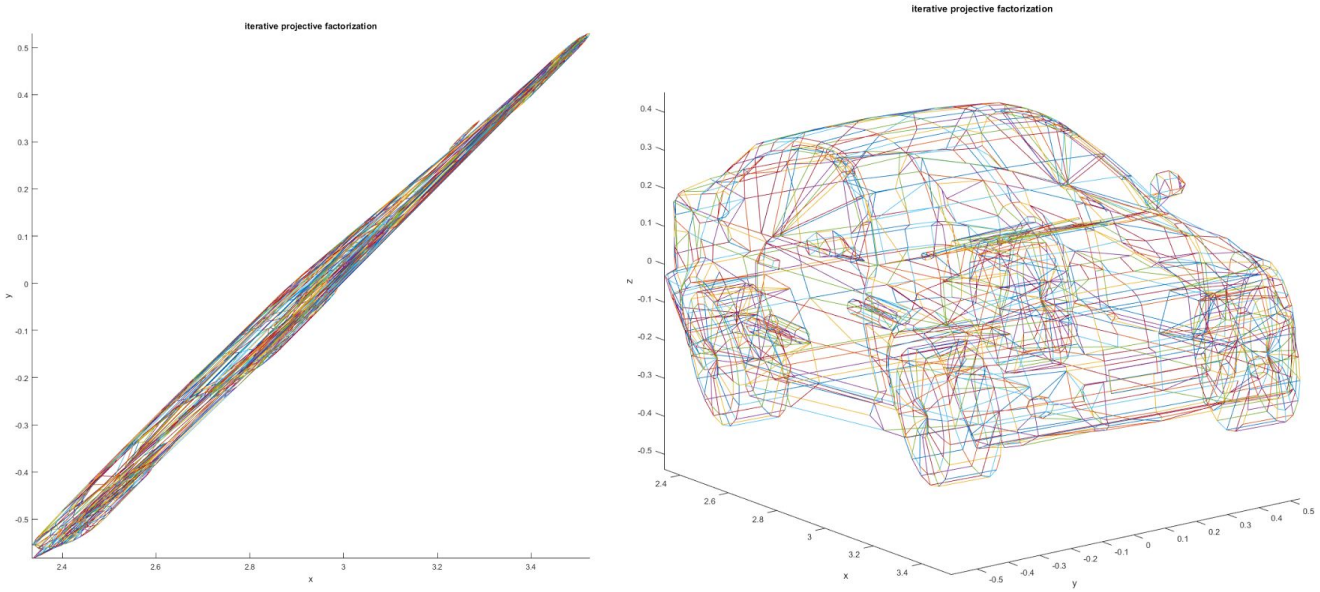

Fig. 3. Visualization of the projective shape looking unrecognizable (left) but after rotation looks recognizable as 2D projection of the car (right).

Due to difficulty in obtaining a good initial solution of Q for Equation (8) for use with LMA to find least squares solution, it often fails to find a solution, and the resulting Q and decomposed A matrices do not invert the projective ambiguity, so the $H^{-1}x$ term is still a projective shape. This problem is shown in Fig. 4 below.

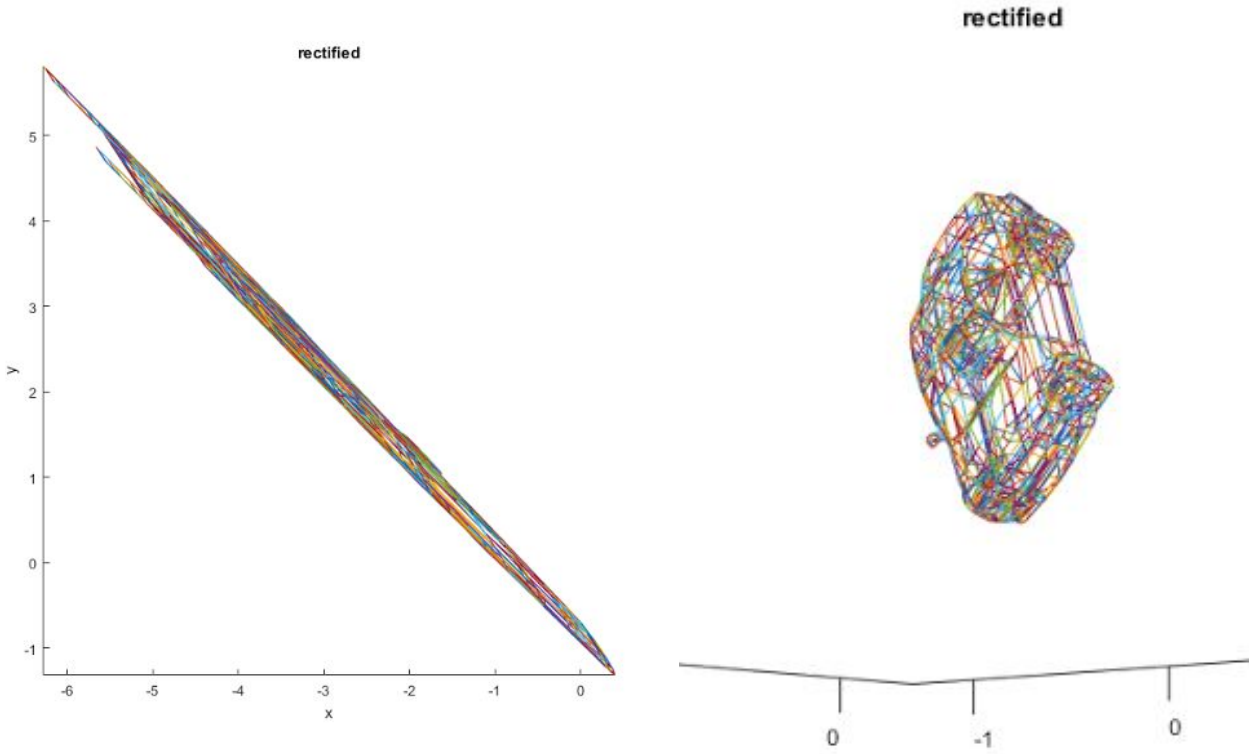

Fig. 4. Error in rectification step still results in projective ambiguity (left), and notice scaling factor applied (right).

Since the result of reconstruction is a similarity transformation to actual object, the error can be evaluated using angles between points and centroid which makes it invariant to uniform scale, rotation and translation. The error is calculated using Equation (41), where x_i are the reconstructed points and d_i are the corresponding ground truth points.

$$ax_j=\text{acos}((x_j-\text{average}(x))\cdot(x_{j+1}-\text{average}(x))/|(x_j-\text{average}(x))|/|(x_{j+1}-\text{average}(x))|)$$
$$ad_j=\text{acos}((d_j-\text{average}(d))\cdot(d_{j+1}-\text{average}(d))/|(d_j-\text{average}(d))|/|(d_{j+1}-\text{average}(d))|)$$
$$\text{error}=[\Sigma_j\text{abs}(ax_j-ad_j)]/\text{col\_count}(x) \quad (41)$$

Using Equation (41), the error of reconstruction is 21° per point pair, which represents a large amount of distortion.

4.2. 3D Shape Registration

The algorithm described in Section 3.3 is implemented and tested synthetically generated curves to obtain quantitative results. List of curves are randomly generated as reference data. A random subset of the reference curves is selected, and a random similarity transformation is applied to simulate the similarity ambiguity from 3D reconstruction. Noise is also optionally applied to each point to simulate image noise in the original input to reconstruction.

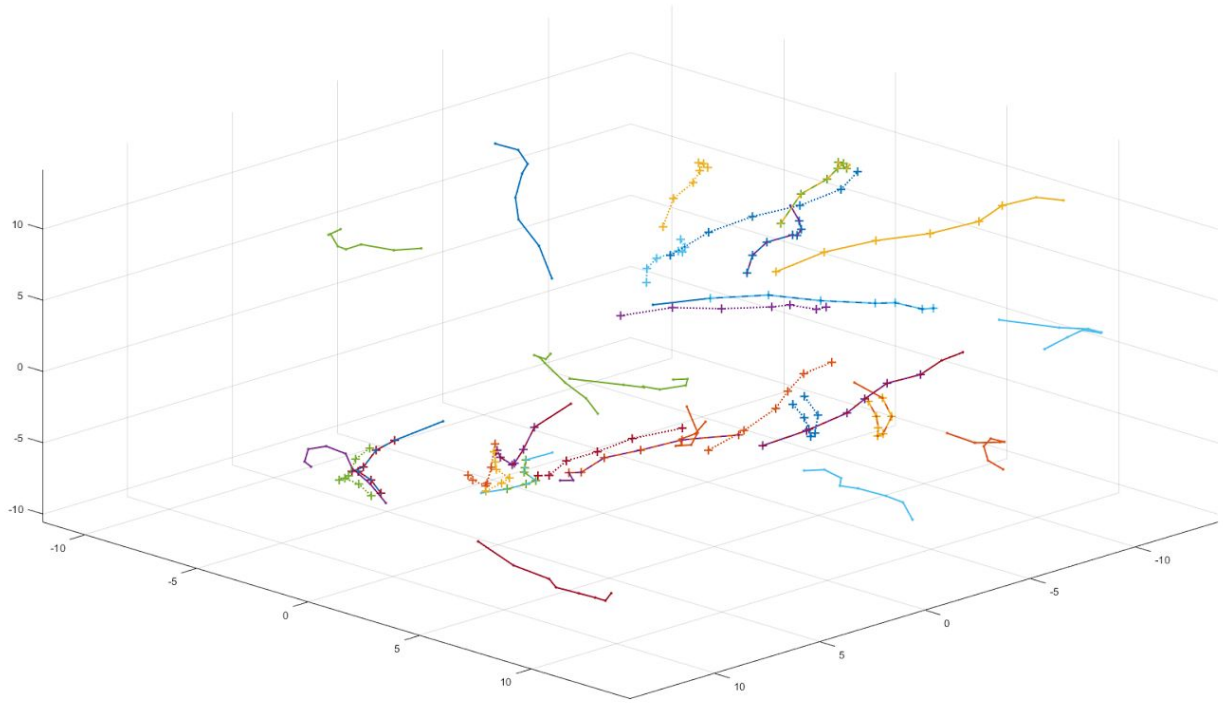

Fig. 5. Visualization of reference curves as solid lines, randomly global similarity transformed test curves as dotted lines, and + markers shows test curved transformed using found similarity transformation solution to match reference curves.

The best case with no noise applied achieves 90.0% success in finding matches, with 1.2×10^{-7} error in the similarity transformation parameters.

With 0.01 noise applied to each point, 0.2 noise applied as non-uniform scale factor, and 2 noise applied as translation, the success rate drops to 70.0% and error increases to 4.1919×10^{-3} which is half of the noise in each point. While keeping these noise and ambiguity parameters the same, noise is added to the ambiguity rotation angles, and result is shown in table below.

Euler Angle Noise (Degrees)	Match Success Rate	Similarity Parameter Error
18	37.5%	4.8598×10^{-3}
12	50.0%	4.9046×10^{-3}
9	77.5%	5.0971×10^{-3}
6	70.0%	5.1822×10^{-3}
3	80.0%	4.8998×10^{-3}
2	75.0%	5.7061×10^{-3}
1	70.0%	5.0498×10^{-3}
0	70.0%	4.1919×10^{-3}

This shows the need to perform Euler angle sweeps in initializing the angle parameters for using LMA to solve the inverse of the similarity ambiguity transformation.

5. Conclusions

The use of 3D reconstruction and 3D shape registration shows some promise in fine-grained classification. But there are several challenges, such as robustly solving a least squares problem for the inverse of similarity ambiguity transform, and parameters to the inverse of projective ambiguity transform.

6. References

[1] Review of existing car identifcation app requiring photo of back. “An App That Can Instantly Identify Any Car (At Least Half the Time)” http://www.wired.com/2014/12/app-can-instantly-identify-car-least-half-time

[2] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3D Object Representations for Fine-Grained Categorization. In ICCV, 2013.

[3] M. Han, T. Kanade. Creating 3D Models with Uncalibrated Cameras. In WACV, 2000.