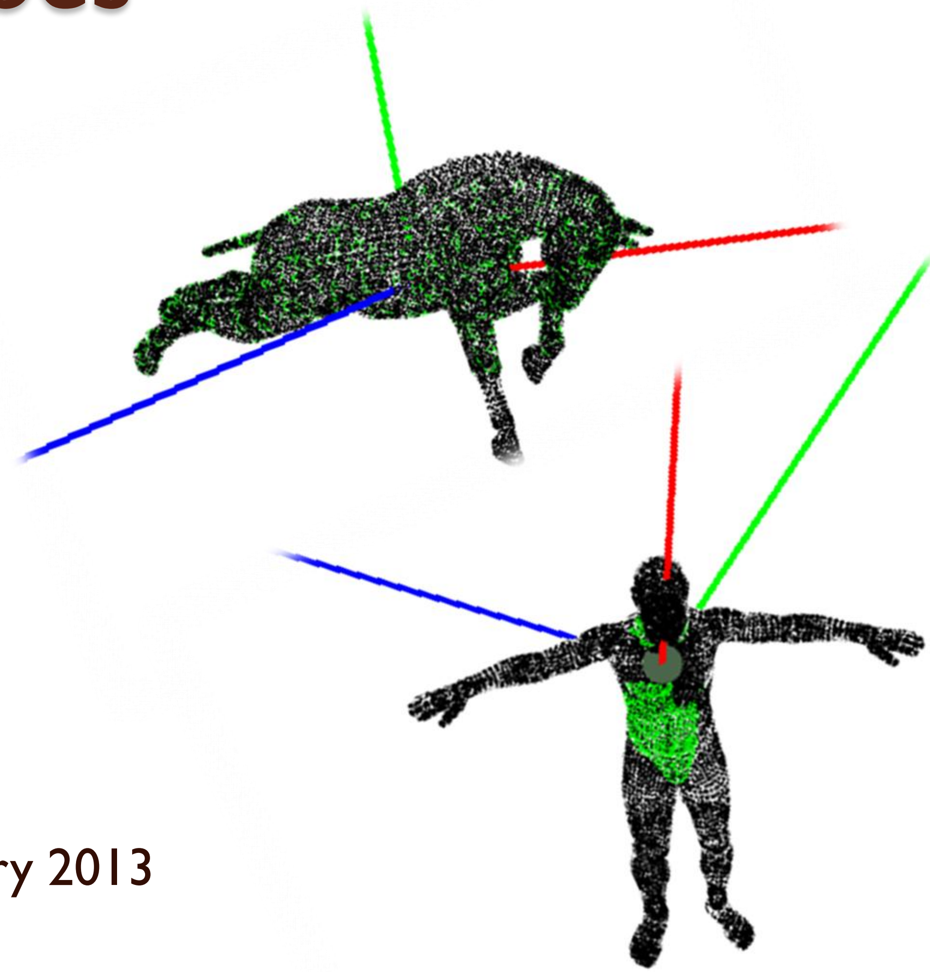
Robust Principal Axes for Point Based Shapes



<u>Presented by:</u>
Deepak Gurung
Lee Yuan Huang

<u>Supervisors:</u>
Yohan Fougerolle
Djamila Aouada

Date: 18th January 2013

Overview

- Objectives
- Introduction
- Theory
- Algorithm Implementation
- Results & Discussion
- Project Management
- Conclusion

Objectives

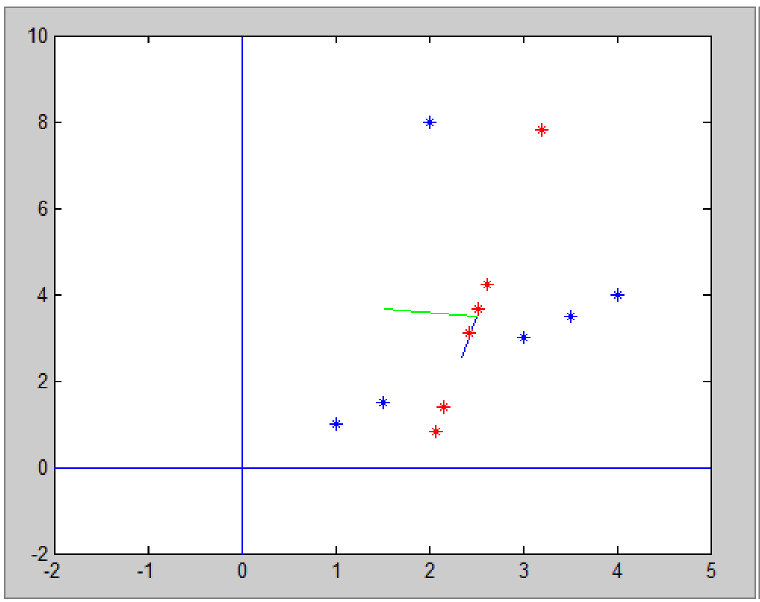
- Implement Robust Principal Axes algorithm in the first paper [1].

Introduction

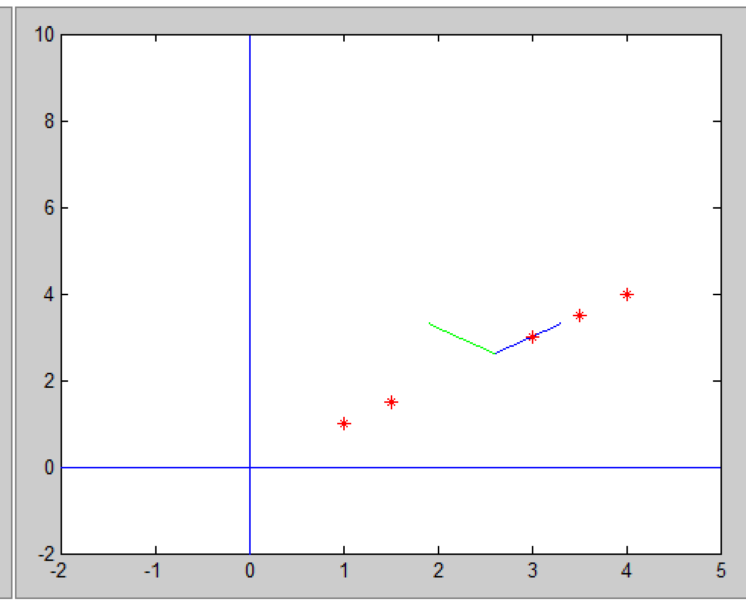
- A more robust way to perform Principal Component Analysis (PCA) on a cloud of points.
- Important to obtain similar Principal Axes even if there are slight changes to the models.
- **Incremental** computation of PCA

Introduction

- Classical PCA – Heavily affected by outlier



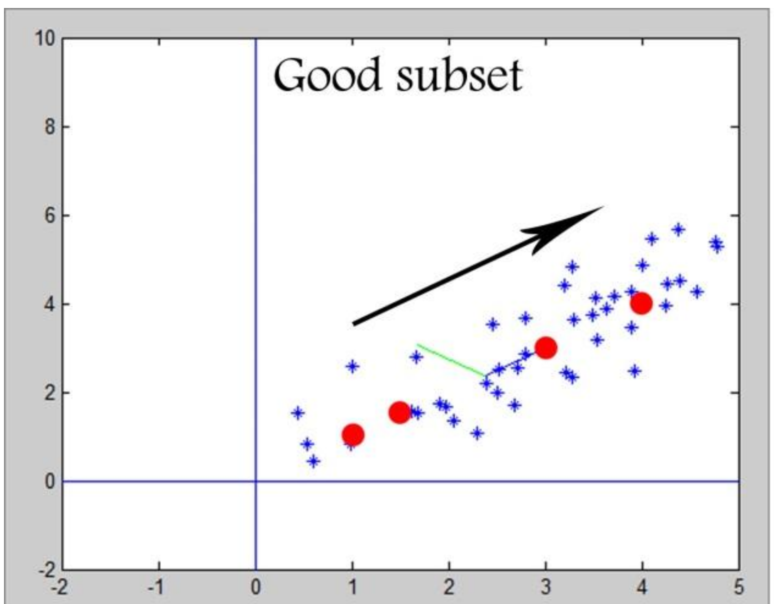
With outlier

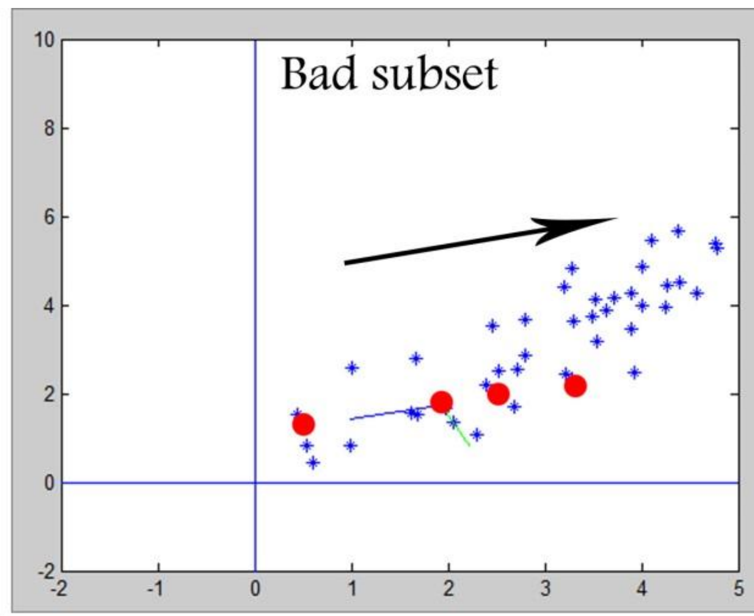


Without outlier

Introduction

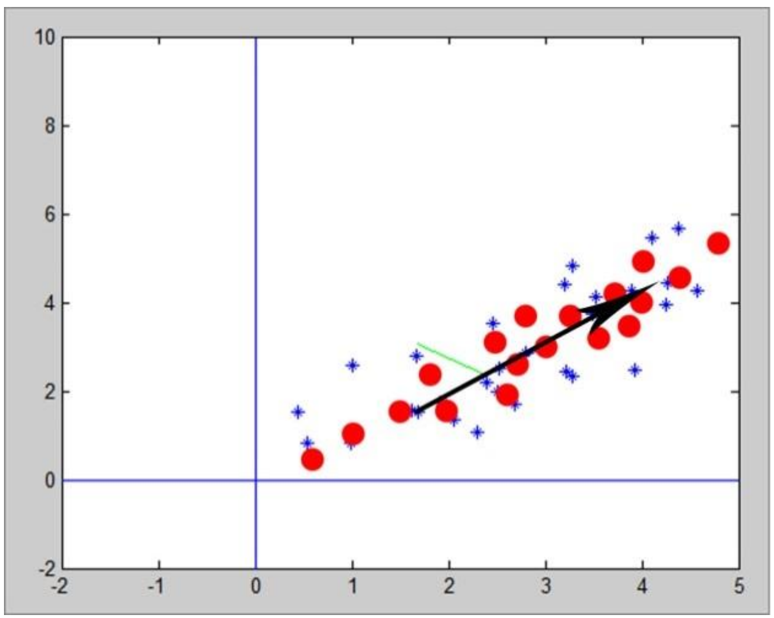
- Robust Statistical Method
- Median – 50% breakdown
- Outlier-free initial subset

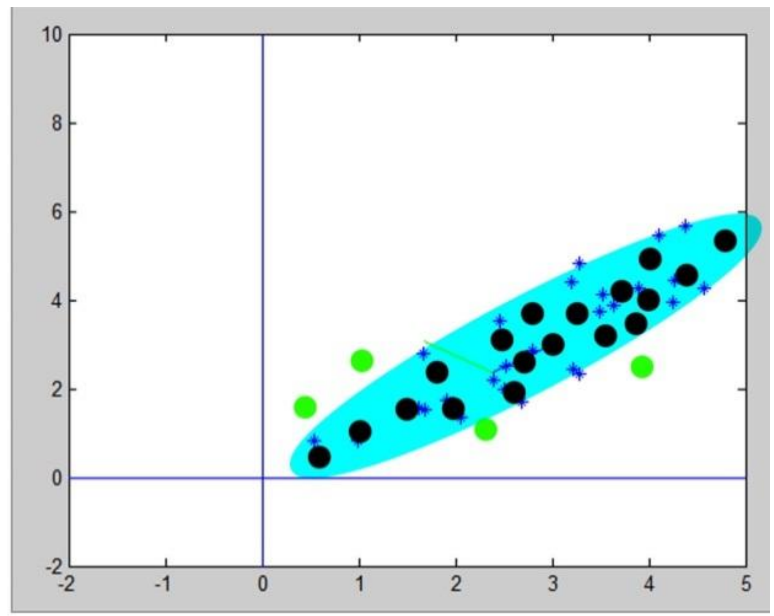




Introduction

- Incrementally increase the subset
- Obtain major region and compute the robust principal axes.





Theory
Principal Component Analysis

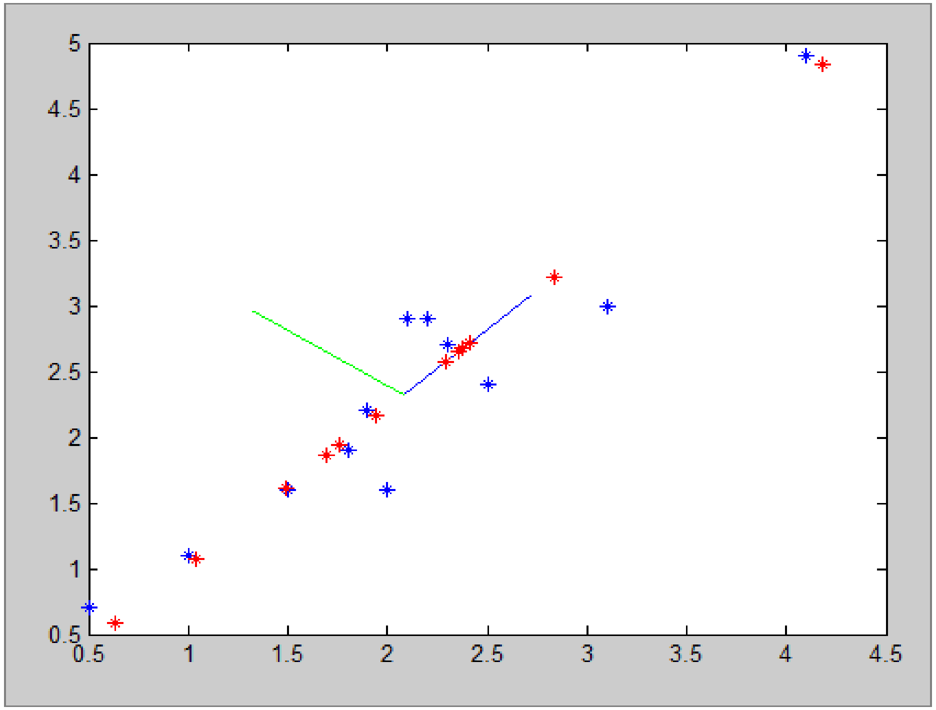
- Very useful mathematical method to view the essence of a complex set of data.

- Principle components can describe the data in simpler forms.

- Goal is to obtain the most meaningful basis to represent our data set

Theory

- 2D example, first principal component represents the highest variance in the data.



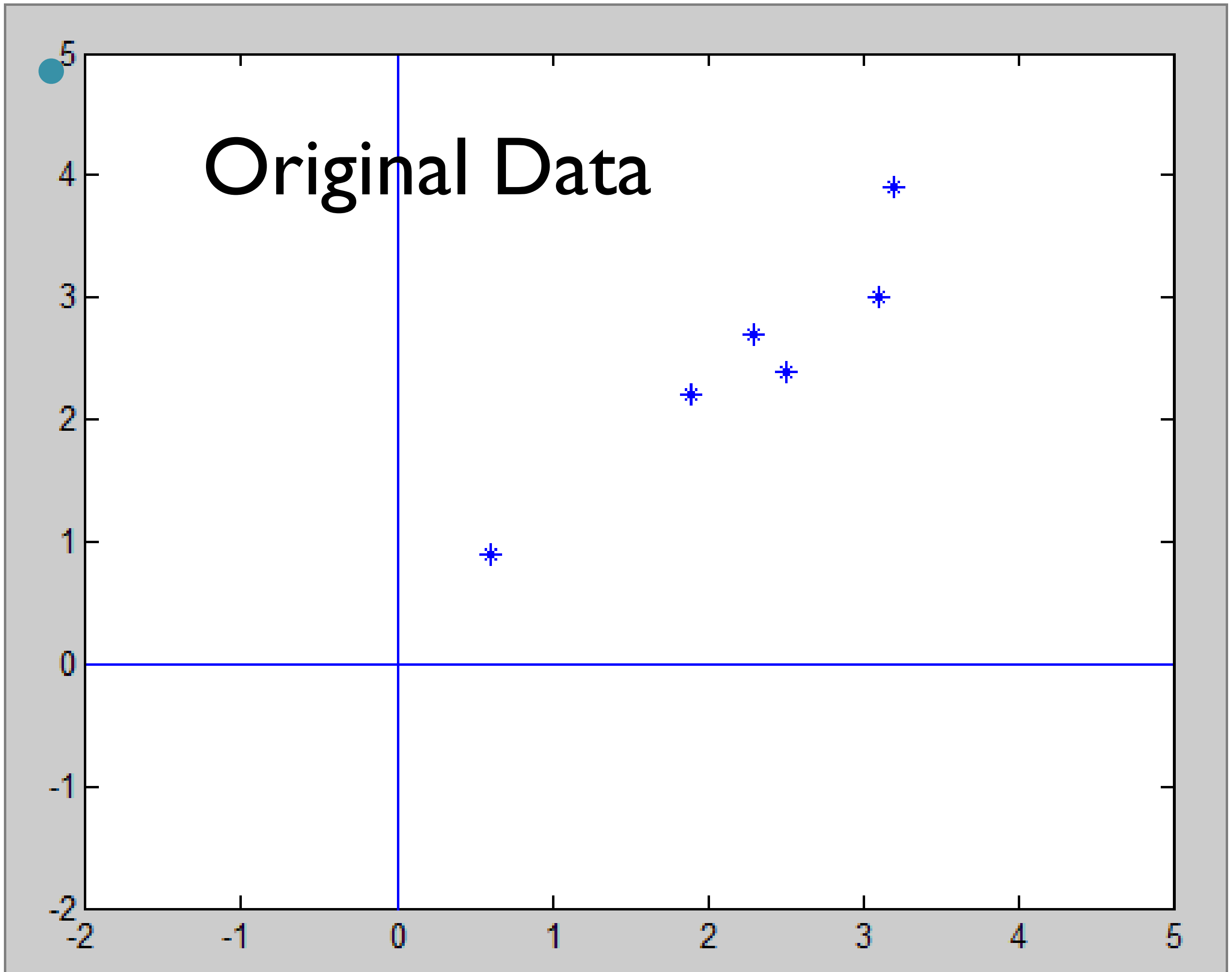
Theory

- Select a normalized direction in m-dimensional space along which the variance in X is maximized. (p1)
- Find another orthogonal direction along which variance is maximized (p2)
- Repeat this procedure, subsequently finding p3, p4… pm. (until m vectors are selected)

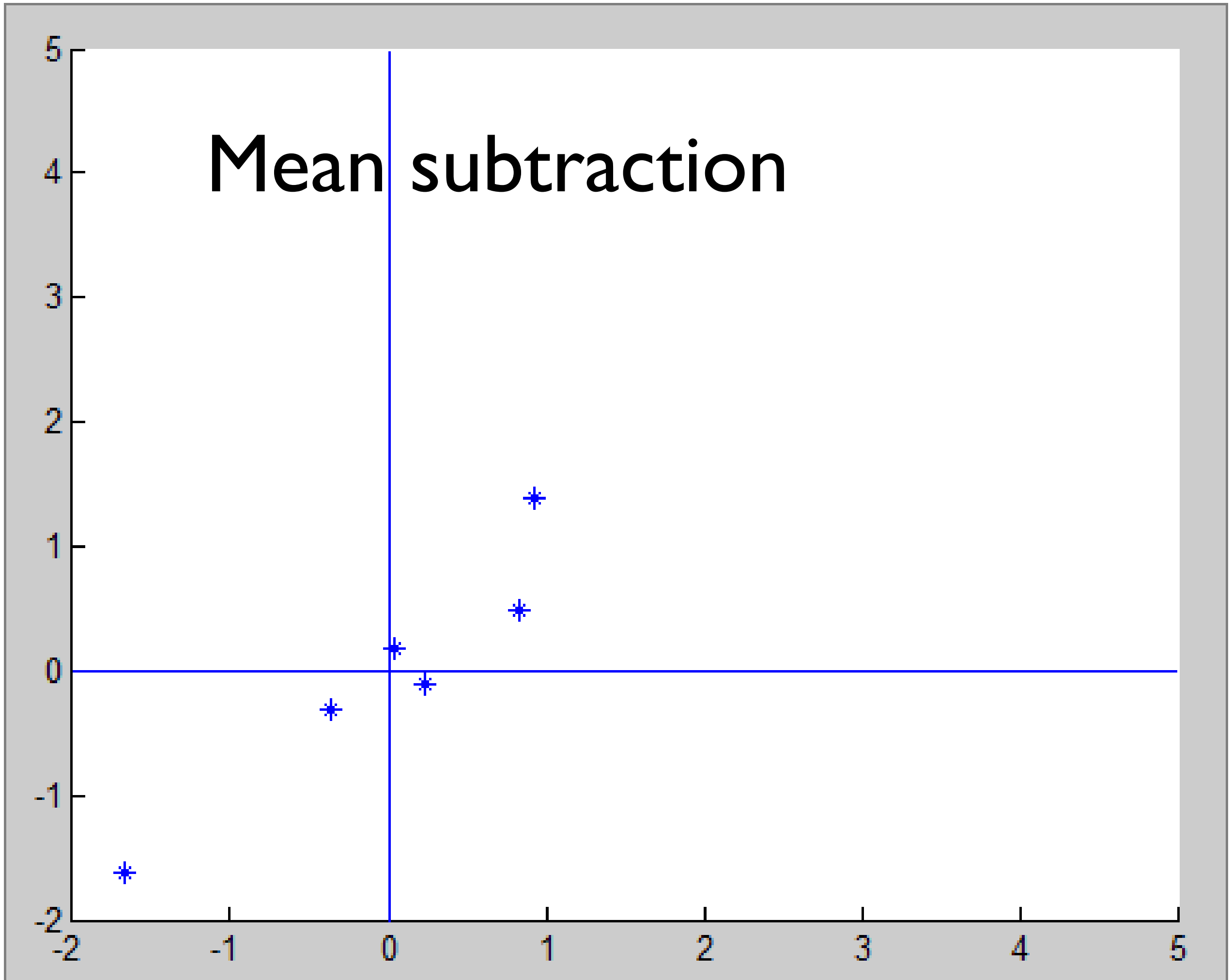
Theory

- Computation of covariance matrix
- $cov(x, y) = \frac{1}{n-1} \sum_i x_i y_i$
- Solving using eigenvectors.
  Principal components = Eigenvectors of covariance matrix
- Solving using Singular Value Decomposition

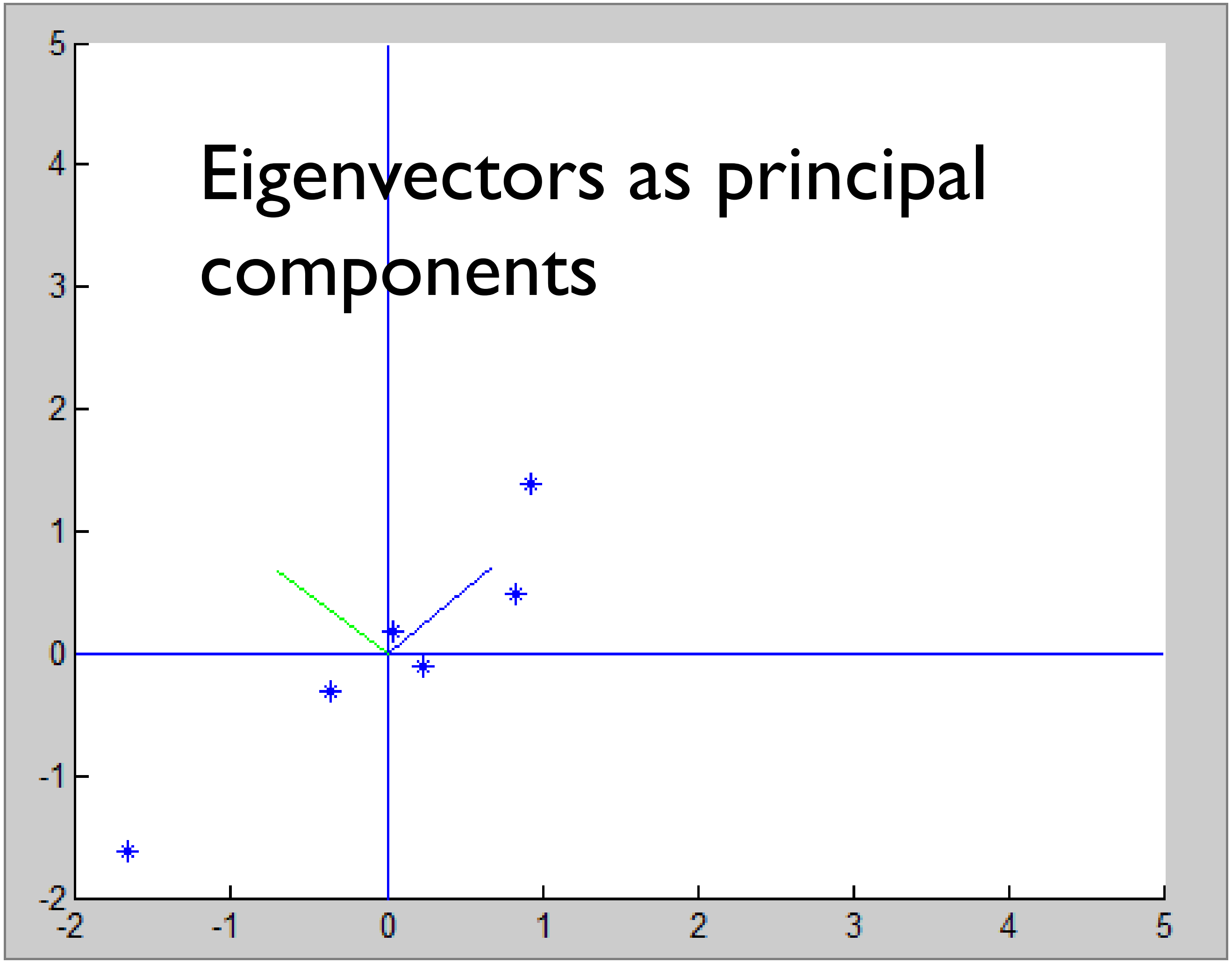
Theory



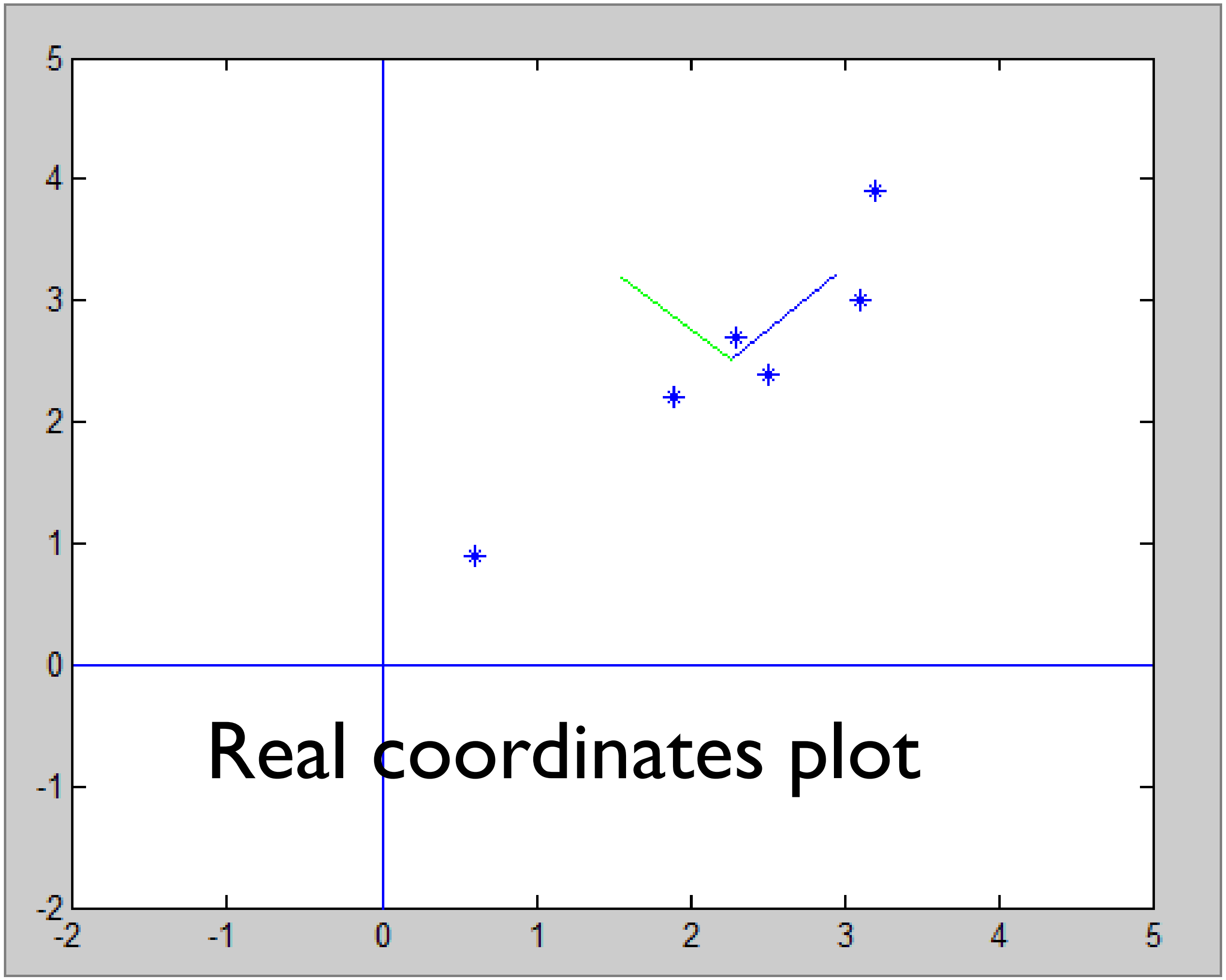
Theory



Theory

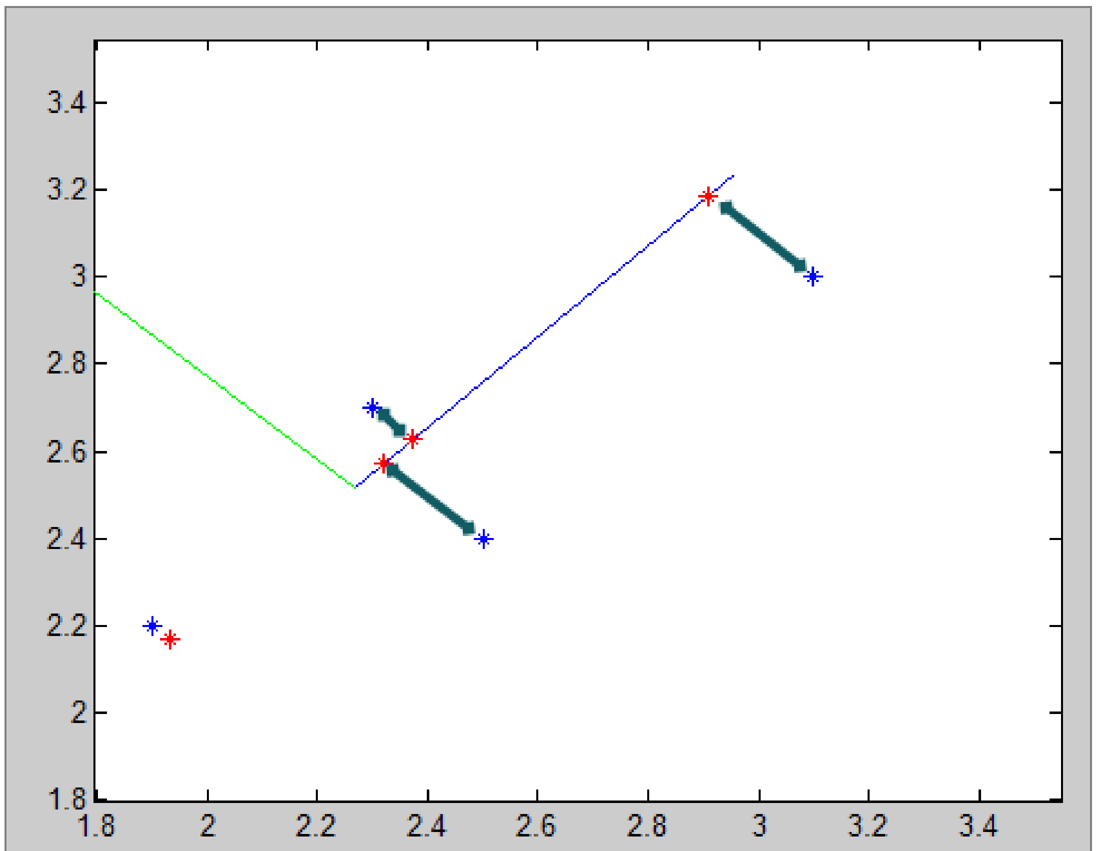


Theory



Theory
Residuals

- Distance between data point and the projected points.



Disadvantage of traditional PCA

- A 3D model may be perturbed with noise and other variables.

- Using conventional PCA does not give correct principal axis as it gives equal weights to all points, inliers and outliers included.

Principal Axes determination using major region.

Main contribution of Liu.

- Separate a model into major region and minor region.
- Major region are the inliers.
- Minor region are the outliers of the model.
- There is no overlap between the points of these two region.
- Use major region to compute principal axes.

Least Median Squares method

- LMS is used to extract the outlier-free major region of a 3D point-based model.
- LMS is used to improve the limitation of PCA.
- LMS is a robust regression method that estimates the parameters of a model by minimizing the median of the absolute residuals.
- LMS has a breakdown of 50%.

LMS problem and its solution

- **Objective function:**

$$residual = \min_{e} \ \underset{i}{median} \ \| (\boldsymbol{o} + \alpha_i \, \boldsymbol{e}) - \boldsymbol{p}_i \|$$

where, $\mathbf{p_i}$ is the test point, **o** is the origin, alpha is scalar quantity and **e** the direction that minimizes the median of residuals.

- **Solution:**
  - RANSAC
    - Need to fix the initial k number of points
  - Forward Search Algorithm
    - Automatically fixes k points

LMS optimization by Forward Search Technique

Steps:

1. Start from a small subset of robustly chosen samples of the data that excludes outliers.

2. Compute principal axes iteratively by adding samples into the initial subset, one at time.

Assumptions for LMS computation

- There is no overlap of major region and minor region.
- The major region contains at least 50% of entire point set.

Step1: Octree based point sampling

- To accelerate extraction of initial outlier-free subset, we use octree.
- Octree with nodes up to a depth level of 5 was created.

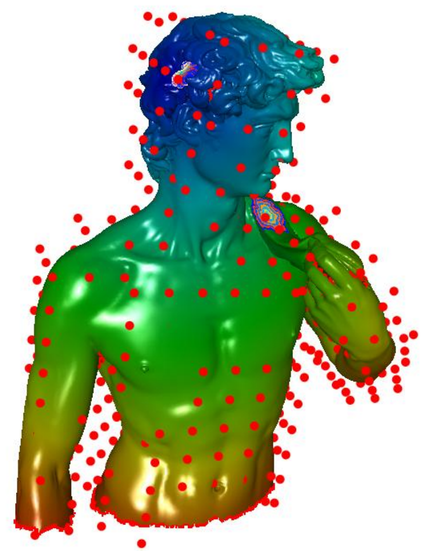


Fig: A model "david" with red points showing the centre of node of an octree.

Step1: Octree based approximation

- Select 4 points for initial subset using octree based sampling.
- Compute principal axes. These are an initial approximation and are fine tuned later on during successive iterations process.

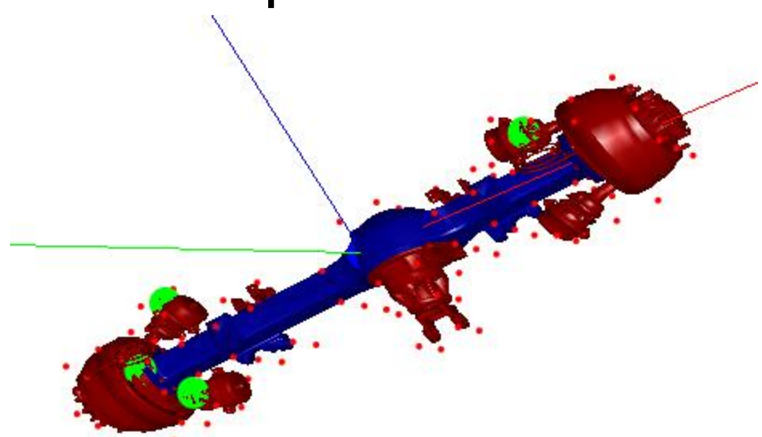


Fig: Four green points denote the initial subset of points.

Step I: Residual band calculation

- Perform octree based approximation for T number of iterations. We chose different values of T.
- The Principal axis with smallest residual for feature points is our initial estimate
- The maximum residual among k points sets the residual band.

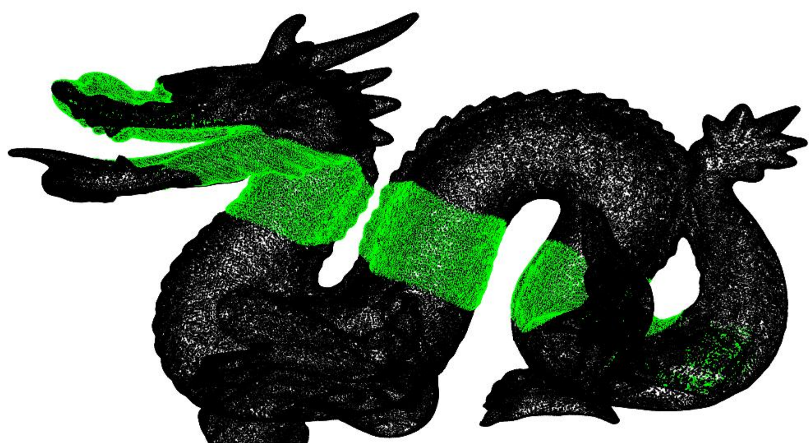


Fig: Green mark is the cylindrical shape of residual about the first principal axis.

Step II: Forward search of major region

- Search and include points which has residuals within the residual band.
- Search and inclusion could be one point at a time or ‘m’ multiple points at a time.
- We chose, m = 100 for higher number of points and m = 30 for less.

Algorithm Implementation

- **PCA**
- Using Eigen library
  - Compute mean
  - Adjust data set
    a. `adjusted_mat = pca_mat.rowwise() - vec_mean.transpose();`
  - Compute SVD
    a. `JacobiSVD <T> svd(adjusted_mat2, ComputeThinU | ComputeThinV);`
  - The columns of matrix V obtained contains the principal axis that we want

Algorithm Implementation

- **Residuals**
- Adjusted data points are projected to the first principal component.
- Shifted to the world coordinate.
- Calculate the residuals as the distance between projected point and data point.

Algorithm Implementation

- **Octree**
  - **Building:**
    - Recursively built an octree.
    - Maximum depth of 5 for our data.
  - **Traverse:**
    - Recursively traverse the octree to access a point
  - **Tabulate points:**
    - Create table of feature points and cumulative density.

Algorithm Implementation

- **Octree (continued)**

CDF	Points of an octree
30	(12, 13, 4) ,(12, 32, 5), …
100	(123, 153, 42), (102, 132, 53), …
150	(232, 123, 80), (243, 132, 55), …
. . .	
20120	(542, 523, 433), (562, 432, 521), …
20999	(700, 643, 423), (742, 562, 525), …
21212	(900, 138, 853), (942, 132, 895), …

Feature points of leaf node
(12, 15, 8)
(125, 140, 45)
(220, 120, 70)
. . .
(550, 450, 500)
(720, 580, 500)
(900, 138, 850)

- **Randomly sample point in octree**

Step I:
Generate random number between
[0 - CDF] and
search the interval for that number.
(Use binary search)

Step II:
After locating the row select a point randomly.

Algorithm Implementation

- Flow chart

Start

↓

Load .off files and initialize parameters/variables.

↓

Mesh is built using the given library.

↓

LMS is executed to obtain the ***k*** random points

Algorithm Implementation

- Flow chart (cont')

Octree of depth **d** is constructed

↓

k points are randomly selected to make up $\mathbf{C}_{\mathbf{temp}}$

↓

PCA is computed on the subset $\mathbf{C}_{\mathbf{temp}}$ to obtain $\mathbf{e}_{\mathbf{1}}$ and **O**

↓

Residuals are computed for each feature point of the boundary cells and are stored in a set. Extract $\mathbf{r}_{\mathbf{half.}}$

↓

if $\mathbf{r}_{\mathbf{half}} < \mathbf{r}_{\mathbf{min}}$

$\mathbf{r}_{\mathbf{half}}$ is stored in $\mathbf{r}_{\mathbf{min}}$ and $\mathbf{C}_{\mathbf{temp}}$ is stored in subset **Q**

Repeat **T** times

Algorithm Implementation

- Flow chart (cont')

Compute $\mathbf{r}_{\mathbf{max}}$ from the $\mathbf{r}_{\mathbf{t}}$.

↓

$\mathbf{C}_{\mathbf{REM}}$ is obtained by removing existing points in **Q** from $\mathbf{C}_{\mathbf{N}}$ using binary search

↓

Compute residuals for points in $\mathbf{C}_{\mathbf{REM}}$.
Store $\mathbf{C}_{\mathbf{REM}}$ in sorted multimap, key = residuals of the point.

↓

Add first **m** points into **Q**.
Maximum residual of **m** points is > $\mathbf{r}_{\mathbf{max}}$?

↓

END

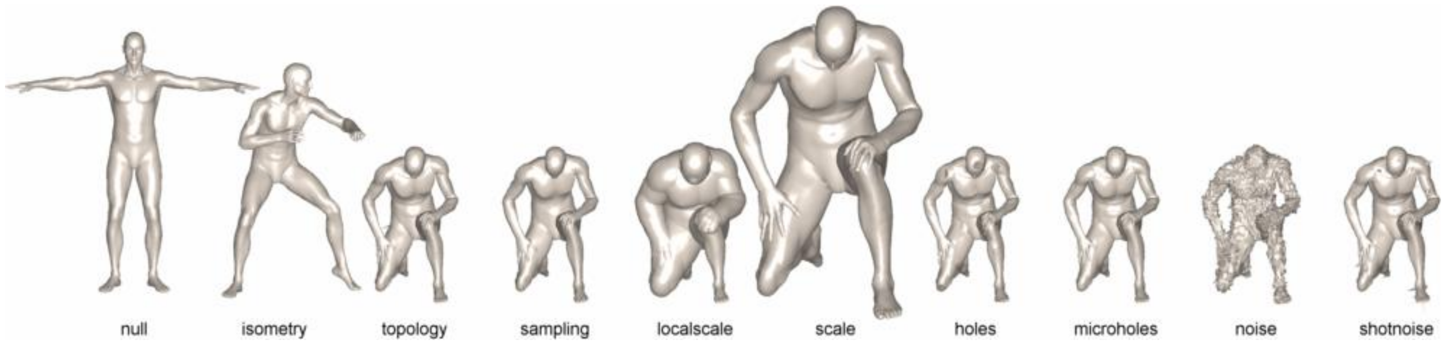
Repeat **MAX_ITERS** time

Algorithm Implementation

- **Tools**
  - Eigen Library
  - Mesh Library
  - OpenGL
  - SHREC 2010 data set
  - Qt

Algorithm Implementation

- SHREC 2010 dataset [2]



- .off files
  contains coordinates of points
  index of points to a triangular face

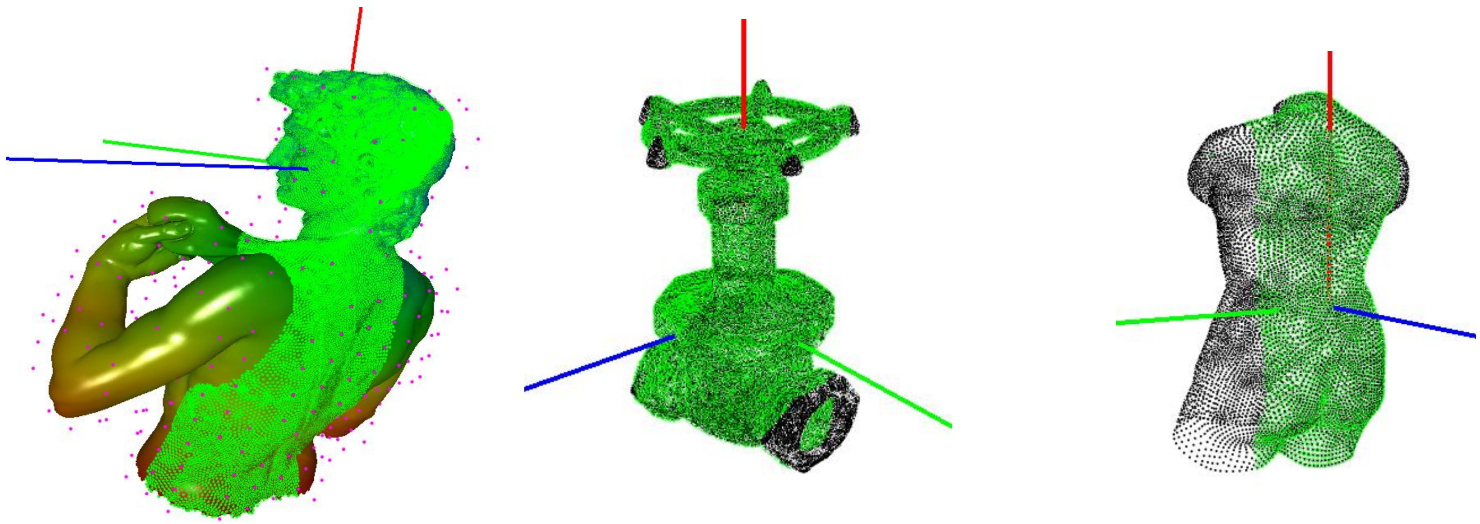
SHREC 2010: Transformations

- **Isometry:** Non-rigid almost inelastic deformations
- **Topology:** Welding of shape vertices resulting in different triangulation
- **Sampling:** The points are down sampled.
- **Local scale:** Local parts of the models are scaled
- **Scale:** Global scale, overall model is resized
- **Holes:** Big holes in the model.
- **Microholes:** Smaller holes in the model.
- **Noise:** Additive Gaussian noise to the points.
- **Shot noise:** Poisson distributed noise. This is also known as Poisson noise.

Results:

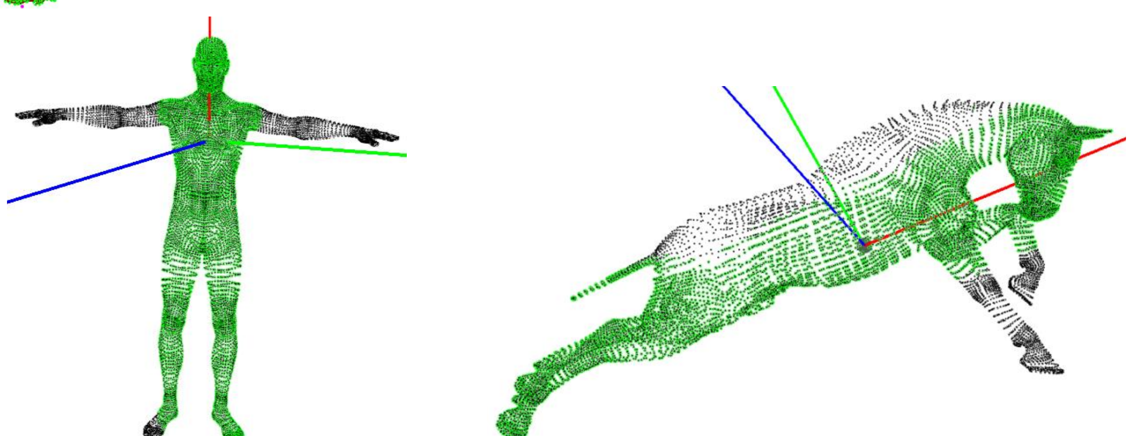
Table 1. Results of Robust Principal axes algorithm for point based models.

Model	Fig	N	Major %	m	T1(s)	T2(s)
david	a	121157	70.1	100	0.36	45
plumbery	b	125434	61.23	100	0.39	50
venus	c	11217	50	30	0.02	1.18
man	d	52565	59.9	30	0.15	18.29
horse	e	19248	80.28	30	0.05	3.4



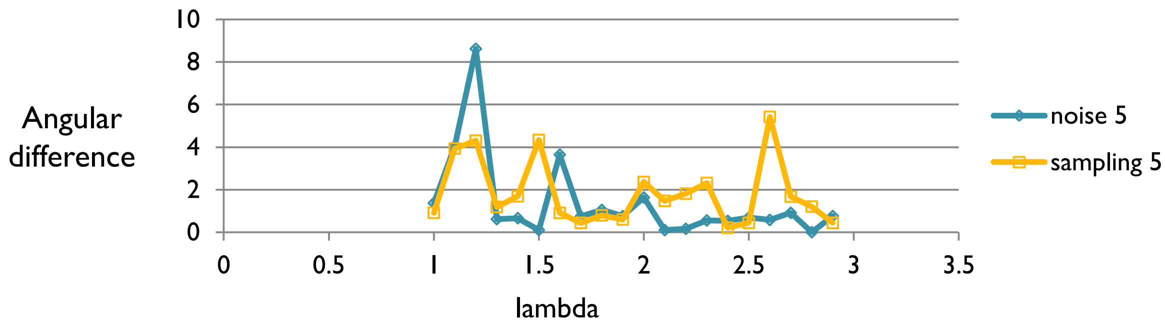
1st PA: Red
2nd PA: Green
3rd PA: Blue

Major region: Green
Minor region: Black



Effect of noise

Angular difference of horse model in Noise 1 Vs noise 5 for different lambda constant determining residual bands.



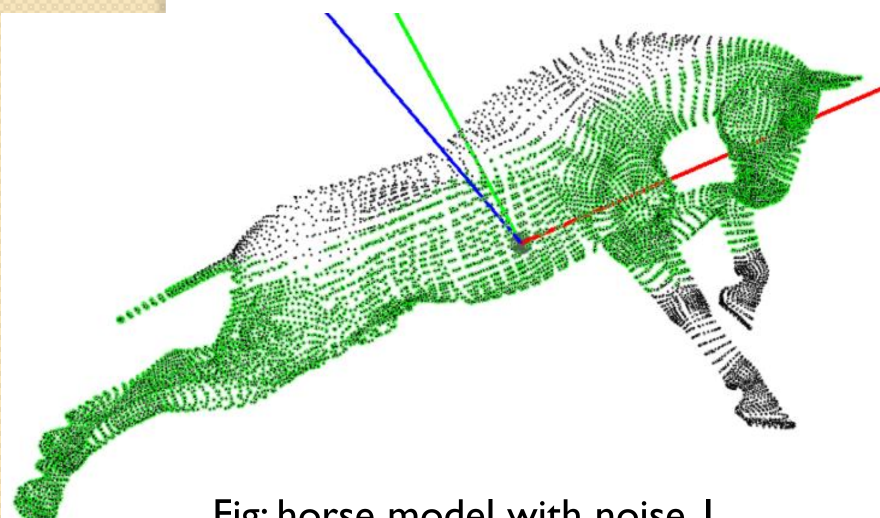


Fig: horse model with noise 1

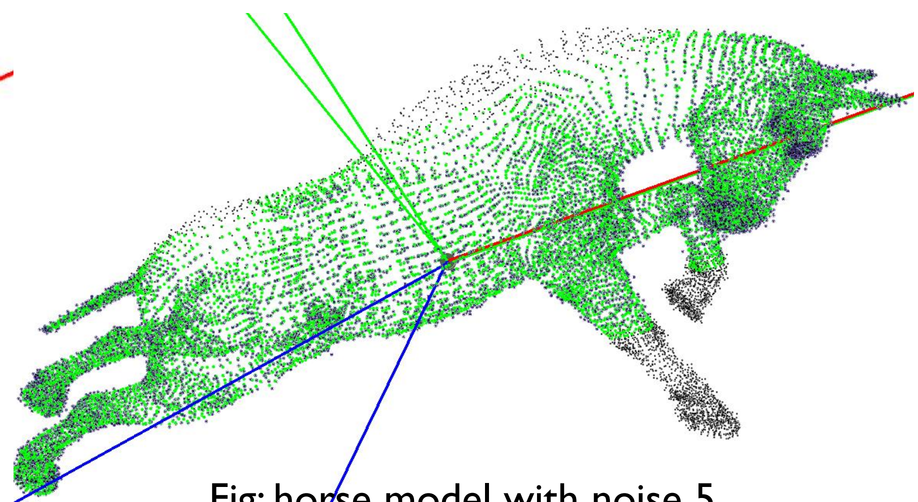
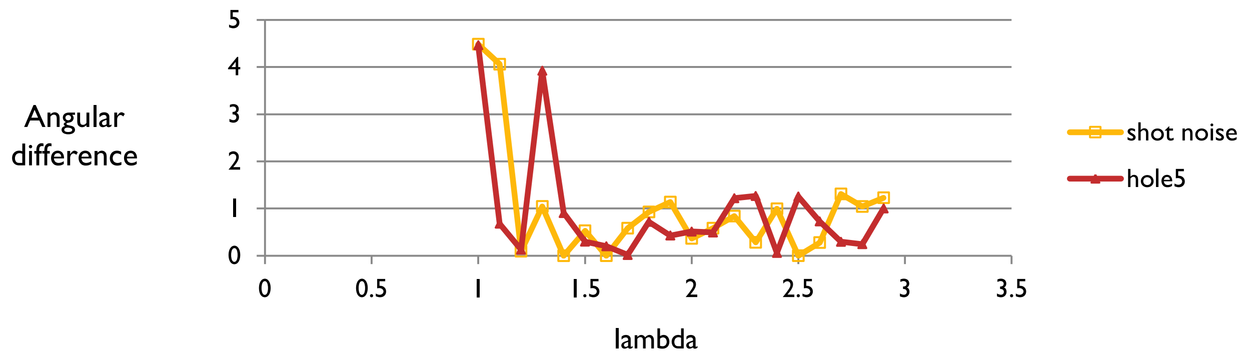
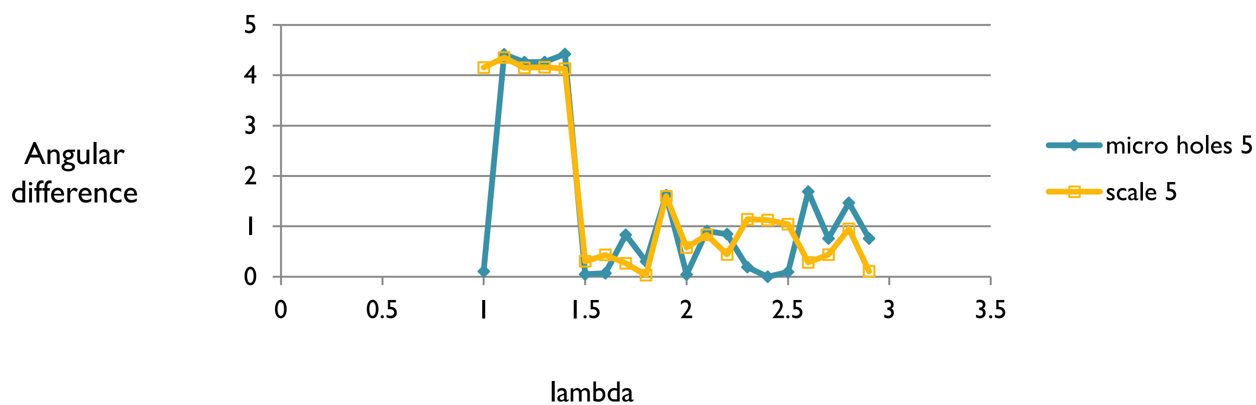


Fig: horse model with noise 5

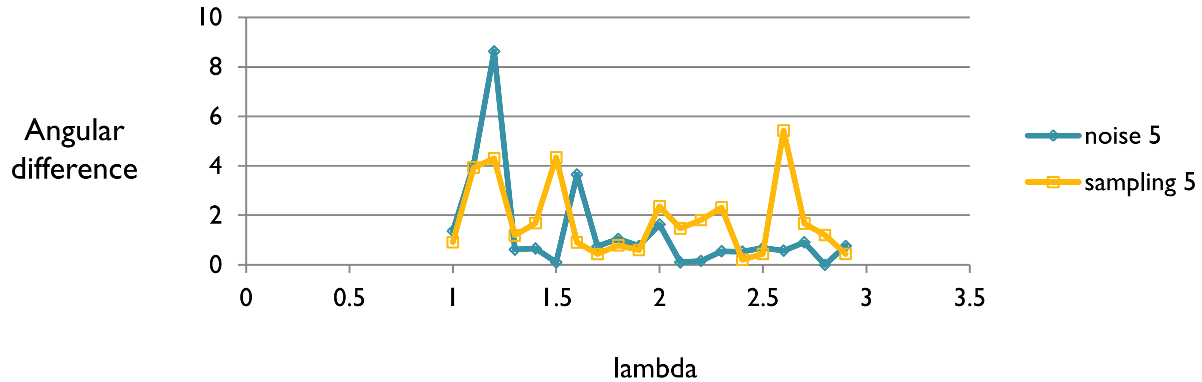
Comparisons of model with least noise and transformed models.

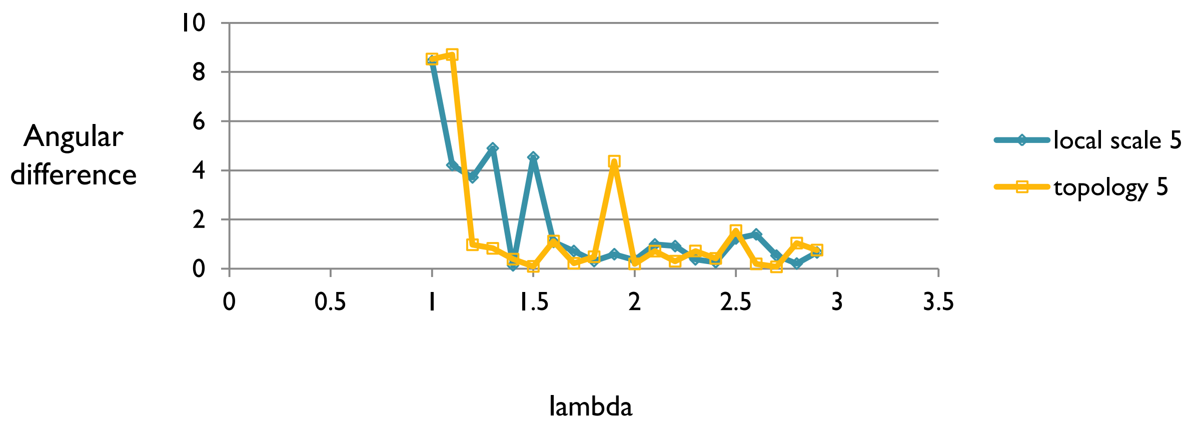
- The graphs below are plot of angular difference between model noise 1 and some known transformations.





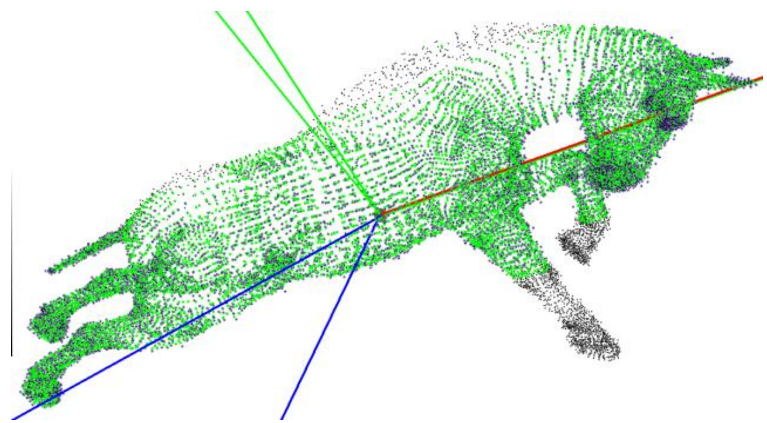
Comparisons of model with least noise and transformed models.





Limitations

- Two main limitations in our project.
  - we computed only the first robust principal axis.



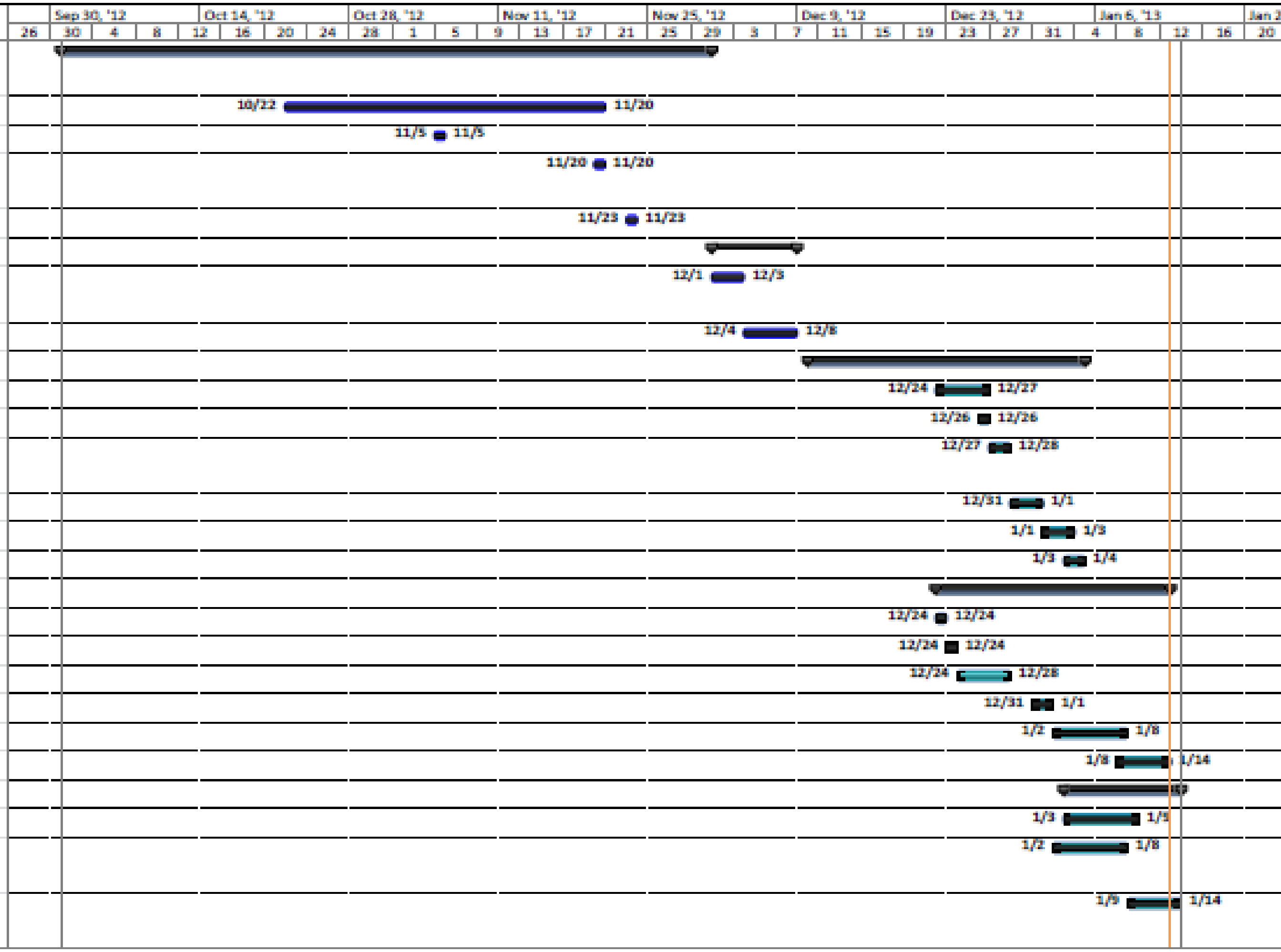
  - We have not made comparison between transformation free model and a transformed model

Project Management

Task Mode	Task Name	Duration	Start	Finish
	Background Research/Reading	**45 days**	**Mon 10/1/12**	**Fri 11/30/12**
	Light reading of materials	22 days	Mon 10/22/12	Tue 11/20/12
	Lectures on Project	1 day	Mon 11/5/12	Mon 11/5/12
	Mini Presentation (Progress)	1 day	Tue 11/20/12	Tue 11/20/12
	Lectures on Project	1 day	Fri 11/23/12	Fri 11/23/12
	Requirements	**5 days**	**Sat 12/1/12**	**Sat 12/8/12**
	Setting up libraries (PCL, Eigen)	2 days	Sat 12/1/12	Mon 12/3/12
	Mesh Library	5 days	Tue 12/4/12	Sat 12/8/12
	Implementation	**20 days**	**Mon 12/10/12**	**Fri 1/4/13**
	Octree Based Sampling	4 days	Sat 12/22/12	Wed 12/26/12
	Least Median Square	1 day	Wed 12/26/12	Wed 12/26/12
	Principal Component Analysis	2 days	Thu 12/27/12	Fri 12/28/12
	Algorithm 2	2 days	Sat 12/29/12	Mon 12/31/12
	Algorithm 1	3 days	Tue 1/1/13	Thu 1/3/13
	Miscellaneous Bugs	2 days	Thu 1/3/13	Fri 1/4/13
	Report Writing	**17 days**	**Sat 12/22/12**	**Sat 1/12/13**
	Drafts/Outline	1 day	Sat 12/22/12	Sat 12/22/12
	Introduction	1 day	Sun 12/23/12	Sun 12/23/12
	Theory/Algorithm	5 days	Mon 12/24/12	Fri 12/28/12
	Implementation Details	2 days	Mon 12/31/12	Tue 1/1/13
	Results/Discussion	5 days	Wed 1/2/13	Tue 1/8/13
	Wrap up	5 days	Tue 1/8/13	Sat 1/12/13
	Testing/Benchmark	**8 days**	**Thu 1/3/13**	**Sun 1/13/13**
	Debugging	5 days	Thu 1/3/13	Wed 1/9/13
	Obtaining results with different models	5 days	Wed 1/2/13	Tue 1/8/13
	Try to do Shape matching and alignment	4 days	Wed 1/9/13	Sun 1/13/13

*tentative

ID	Indicators	Task Mode	Task Name	Duration
1	✓		**Background Research/Reading**	**45 days**
2	✓		Light reading of materials	22 days
3	✓		Lectures on Project	1 day
4	✓		Mini Presentation (Progress)	1 day
5	✓		Lectures on Project	1 day
6	✓		**Requirements**	**5 days**
7	✓		Setting up libraries (PCL, Eigen)	2 days
8	✓		Mesh Library	5 days
9	✓		**Implementation**	**20 days**
10	✓		Octree Based Sampling	4 days
11	✓		Least Median Square	1 day
12	✓		Principal Component Analysis	2 days
13	✓		Algorithm 2	2 days
14	✓		Algorithm 1	3 days
15	✓		Miscellaneous Bugs	2 days
16			**Report Writing**	**17 days**
17	✓		Drafts/Outline	1 day
18	✓		Introduction	1 day
19			Theory/Algorithm	5 days
20	✓		Implementation Details	2 days
21	✓		Results/Discussion	5 days
22	✓		Wrap up	5 days
23	✓		**Testing/Benchmark**	**8 days**
24	✓		Debugging	5 days
25	✓		Obtaining results with different models	5 days
26	✓		Try to do Shape matching and alignment	4 days



Project: Project1
Date: Sat 1/12/13

Task	Group By Summary	External Milestone	Duration-only	Deadline
Split	Rolled Up Task	Inactive Task	Manual Summary Rollup	Progress
Milestone	Rolled Up Milestone	Inactive Milestone	Manual Summary	
Summary	Rolled Up Progress	Inactive Summary	Start-only	
Project Summary	External Tasks	Manual Task	Finish-only	

Page 1

Conclusion

- We implemented robust principal axes determination of point-based shapes using least median squares (LMS) method.
- We used octree based approximation and point sampling for LMS optimization.
- We extracted major region.
- Finally, we computed first principal axis.

Future Recommendation

- Use the method suggested by Liu to update second and third principal axis.
- Use the algorithm for various applications like, shape alignment, as a preprocessing step of Iterative Close Point (ICP) approximation for 3D surface reconstruction.

Reference

[1] L. Yu-Shen and R. Karthik, "Robust principal axes determination for point-based shapes using least median of squares," Computer-Aided Design, vol. 41, pp. 293-305, 2009.

[2] A. M. Bronstein, M. M. Bronstein, U. Castellani, A. Dubrovina, L. J. Guibas, R. K. R. P. Horaud, D. Knossow, E. v. Lavante, D. Mateus, M. Ovsjanikov and A. Sharma, "SHREC 2010: robust correspondence benchmark," in Proc. EUROGRAPHICS Workshop on 3D Object Retrieval (3DOR), 2010.

Acknowledgements

- .off loader by Juan
- Yohan’s Mesh Library
- Danda Pani Paudel

Questions?