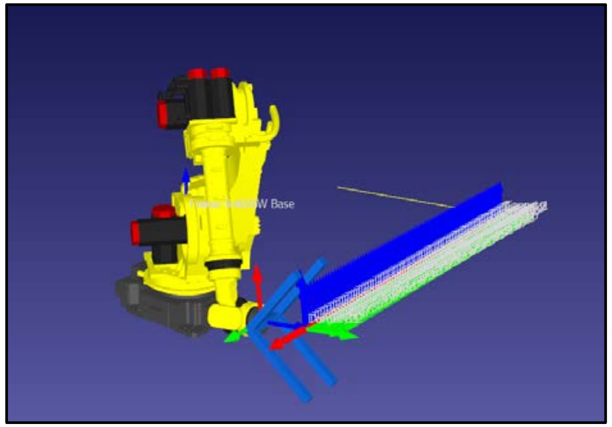
# A Novel Method of Robotic Toolpath Generation With 3D Imaging for Wind Turbine Blade Finishing

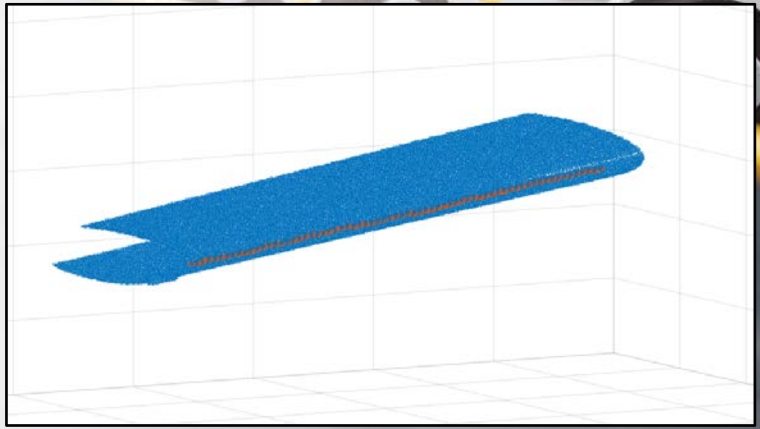


CIRC 2021

**Casey Nichols**

National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, operated by the Alliance for Sustainable Energy, LLC.



# Agenda

- Introduction – Presenter, NREL, IACMI
- Background – Wind Turbine Blade Finishing
- Industrial Robots in Blade Finishing
- Introduction to three-dimensional (3D) Imaging
- Point Cloud Generation and Processing
- Robotic Toolpath Generation
- System Integration
- Conclusion
- Future Work

GE-LM 107-m blade
Source: LM Wind

# Introduction

Meet the team

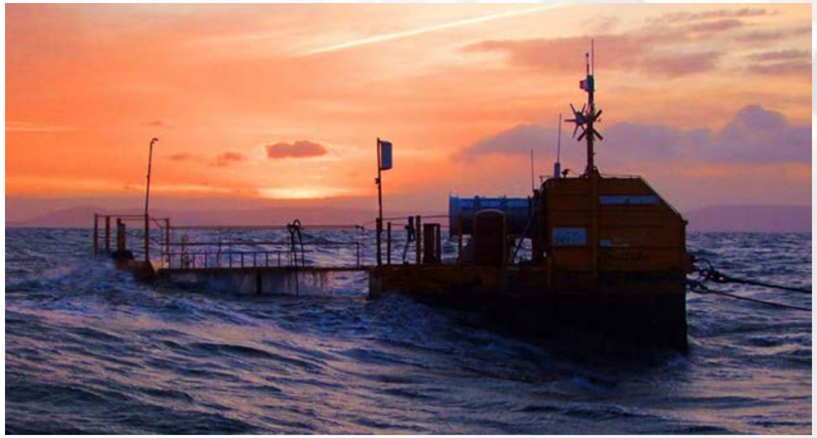
# Presenter – Casey Nichols

Research engineer in wind and marine renewable energy at the National Renewable Energy Laboratory (NREL)

NREL Transforming ENERGY

*NREL advances the science and engineering of energy efficiency, sustainable transportation, and renewable power technologies and provides the knowledge to integrate and optimize energy systems. www.nrel.gov*


By Werner Slocum (NREL) 62958


By Ocean Energy Limited 17873

# Composites Manufacturing Education and Technology Facility (CoMET)

By Dennis Schroeder / NREL 41872

By Dennis Schroeder / NREL 54061

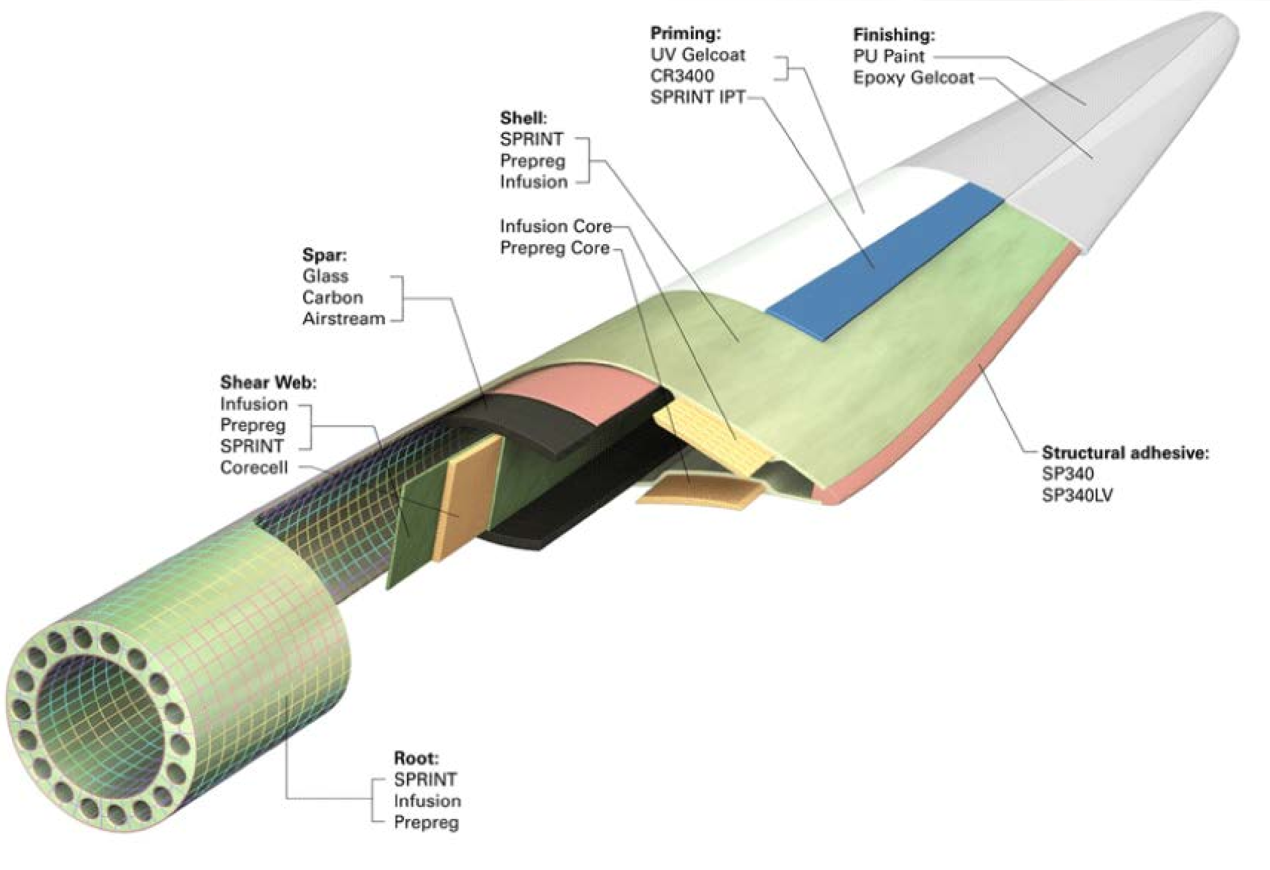
# Project Partners

- GE Renewables and GE Research Center
- LM Wind
- The Institute for Advanced Composites Manufacturing Innovation (IACMI)
- U.S. Department of Energy Advanced Manufacturing Office
- Colorado Office of Economic Development and International Trade

# Background: Wind Turbine Blade Finishing

# Wind Turbine Blade Structure Design

Wind turbine blades are constructed of several composite structures bonded together to ensure high strength and low weight.




Components of a Wind Blade
Source: Gurit

# Wind Turbine Blade Manufacturing Process

1. Each component is infused in separate molds
2. The shear web is bonded to one of the skins in the clamshell mold
3. The clamshell mold is closed with the shear web inside, and then all components are bonded together.

DNV GL Clamshell Molds
Source: North American Wind Power

# Finishing Wind Turbine Blades

Flashing trimming
Source: LM Wind

Grinding
Source: LM Wind

# Achieving Optimal Leading and Trailing Edges

After flashing trimming, the leading and trailing edges have a small ridge that must be ground off to achieve the desired airfoil profile.

Preoverlamination blade root
Source: LM Wind

# Blade Finishing From an Environmental Health and Safety Point of View

Blade finishing includes the following risks:

- Dust explosion hazard
- Respiratory hazard
- Excessive noise injury
- Musculoskeletal injury
- Work from heights
- Fatigue.

Source: Klingspor Windpower


Source: LM Wind

# Industrial Robots for Wind Turbine Blade Manufacturing

Robots in the wind turbine blade manufacturing plant

# Robotic Processes Are Involved in the Manufacturing of Blades but There Are No Major Successes in Blade Finishing

- ABB blade painting solution
- Kuka mobile platform blade polishing


ABB blade painting robots
Source: ABB


Kuka robotic polishing mobile platform
Source: Kuka

# The Challenge: Remove the Uncertainty of Blade Geometry To Enable Robotic Toolpath Following

The root
Source: General Electric

The tip
Source: Vestas

The solution: capture the blade geometry as-built and process the data to determine the robotic finishing toolpath.

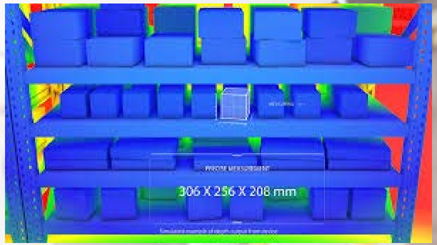
# Introduction to 3D Imaging

Capturing Point Clouds

# 3D Imagining Used to be a High-Tech, Expensive Process but Is Now Available in Easy-to-Use, Low-Cost Packages








Sources: Intel, LUCID Vision Labs

| Camera | Advertised Precision | Frames Per Second (fps) | Field of View | Price |
|---|---|---|---|---|
| Intel Realsense D415 | < 2% at 2 meters (m) | 90 fps | 0.5–3 m | $150 |
| Intel Realsense L515 | 5–14 millimeters (mm) | 30 fps | 0.25 m–9 m | $350 |
| Lucid Vision Helios2 | +/- 4 mm | 30 fps | 0.3–8.3 m | $1,500 |

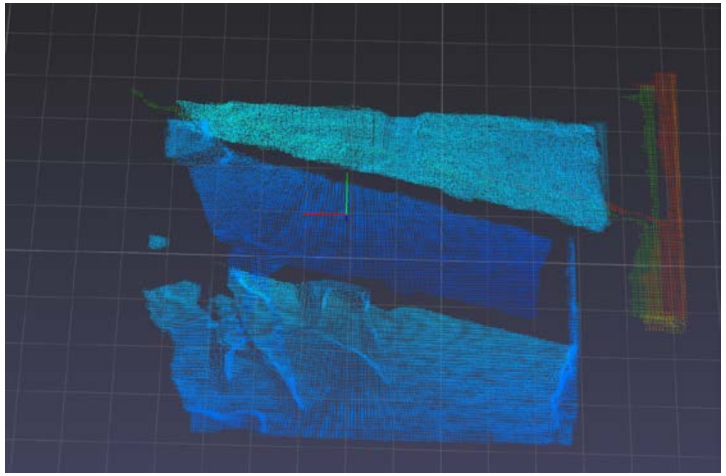
| Selection Criteria |
|---|
| 1 – Precision/Cost<br>2 – Field of View<br>3 – Availability of API |

# 3D Imaging – Stereo Depth vs. Lidar

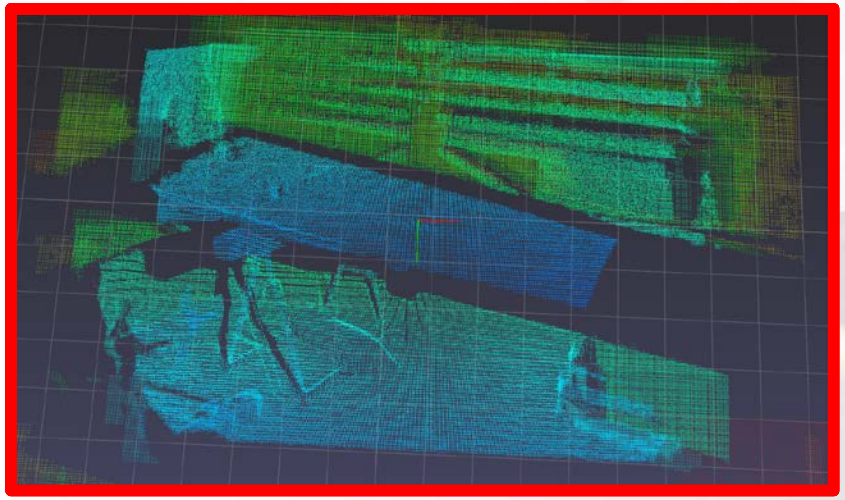
- Stereo depth perception
- Intel Realsense D415

- Lidar imaging
- Intel Realsense L515

Color image



Stereo D415

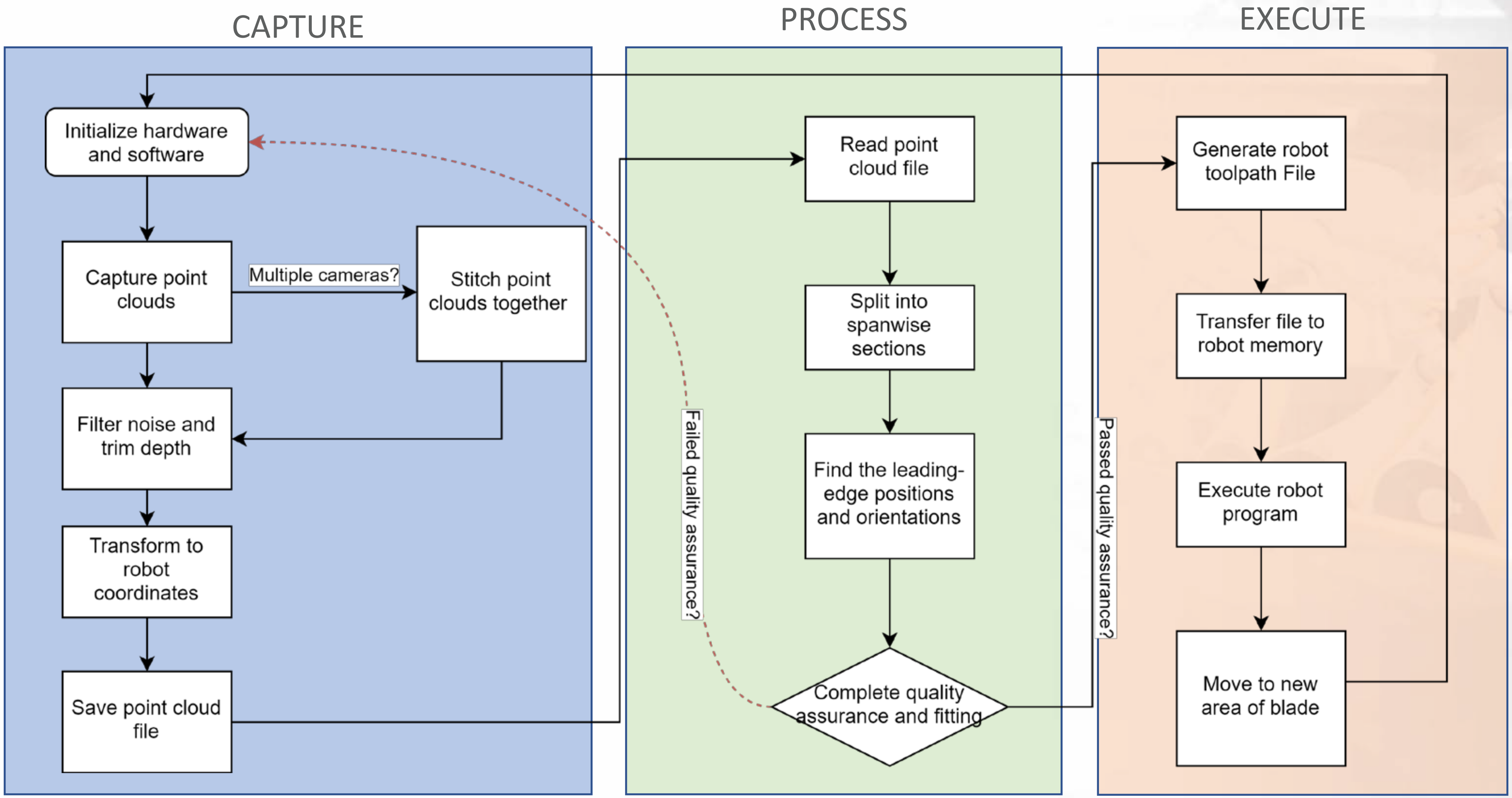


Lidar L515

# Robotic Toolpath Generation

Capturing and Processing Point Clouds

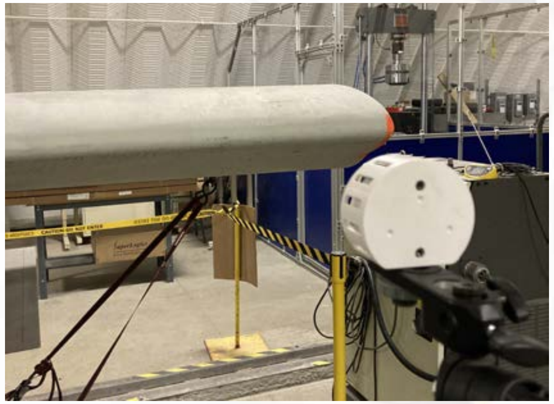
# Procedure Overview



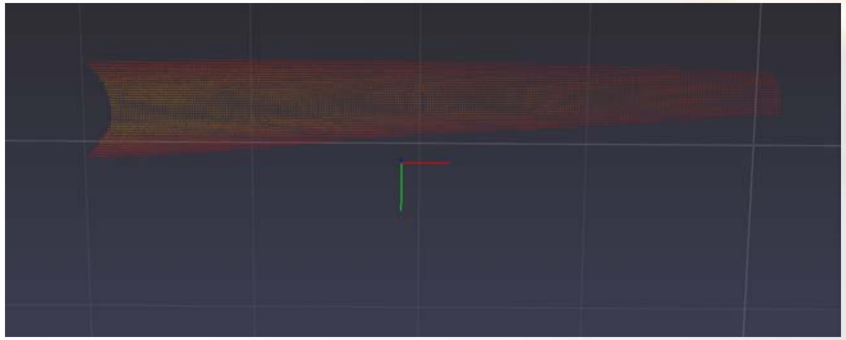
# Capturing the Point Cloud With the Camera

CAPTURE

- Using the Intel Realsense API in Python, the point cloud is captured and stored in memory
- If multiple cameras are used, the point clouds must be stitched together to form one point cloud
- The point cloud is then edited to remove unneeded data
  - Limit depth from camera to remove background geometry
- The point cloud coordinates are based on the camera location so the points must be transformed into robot coordinates.

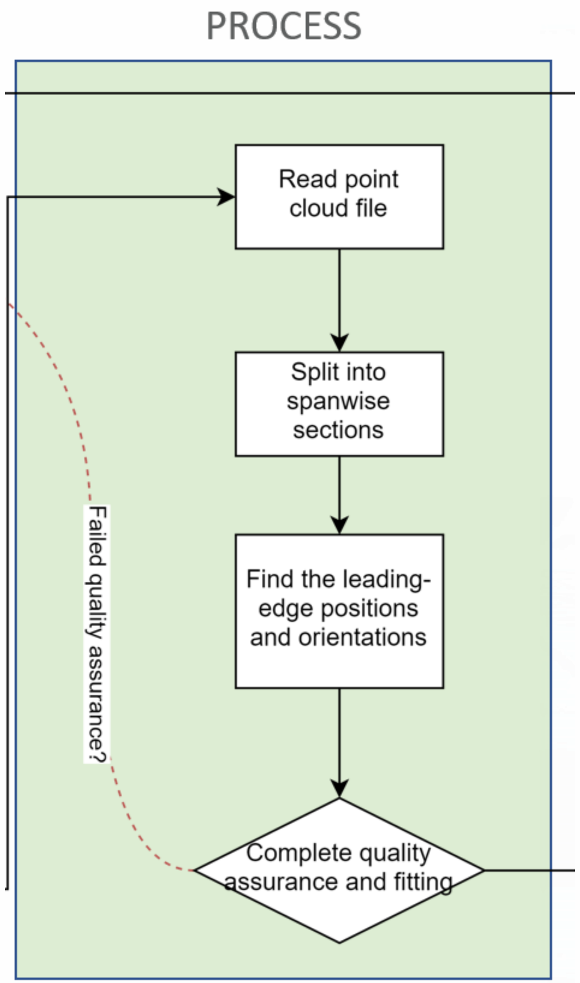


Intel L515 capturing blade section point cloud

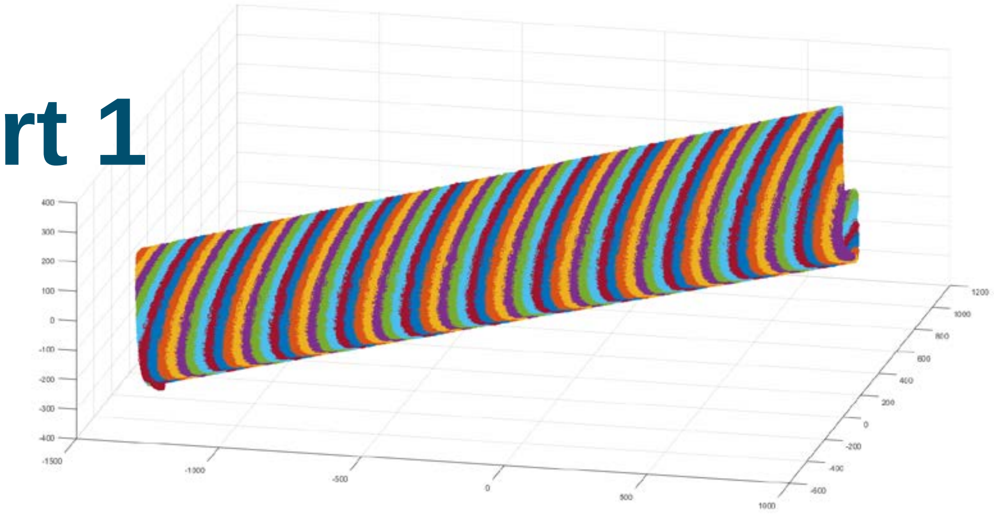


Rendering of leading-edge point cloud

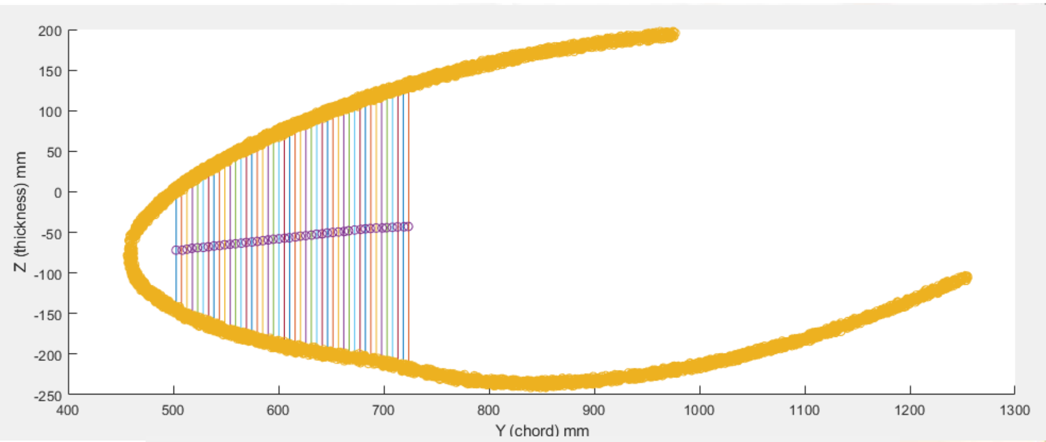
# Point Cloud Processing – Part 1



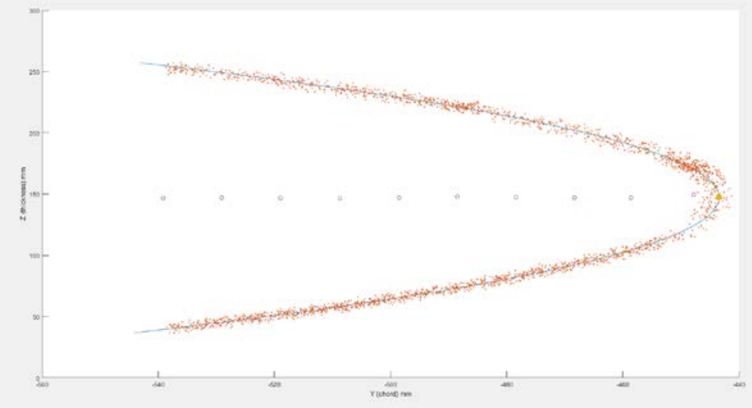

Split point cloud into spanwise sections



Determine twist angle of blade and rotate to achieve parabola shape

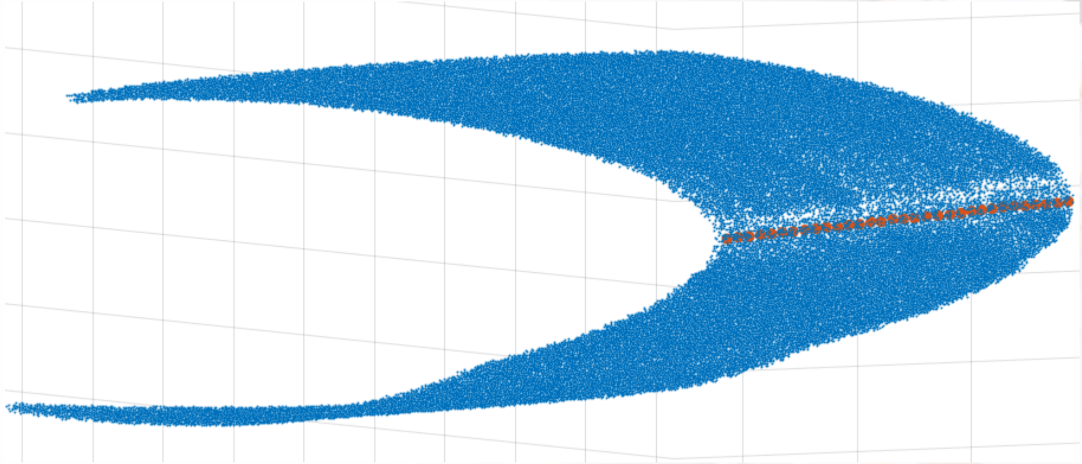



Fit rotated point cloud with second-order polynomial



# Point Cloud Processing – Part 2

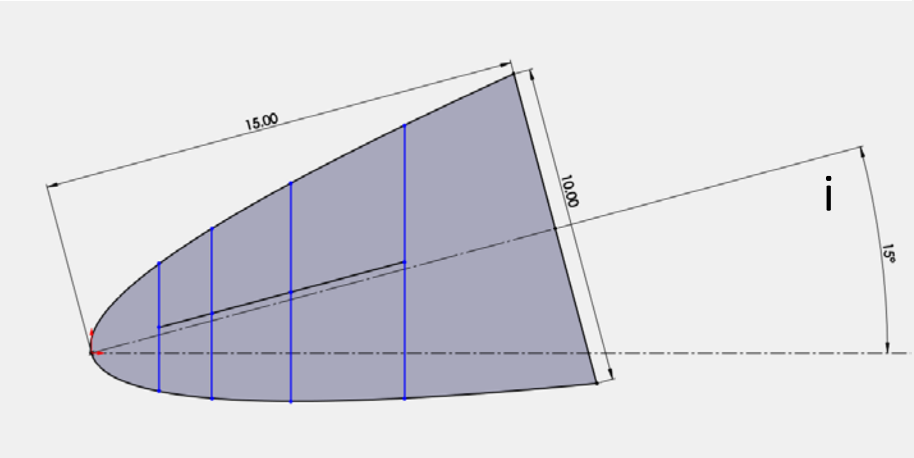
PROCESS

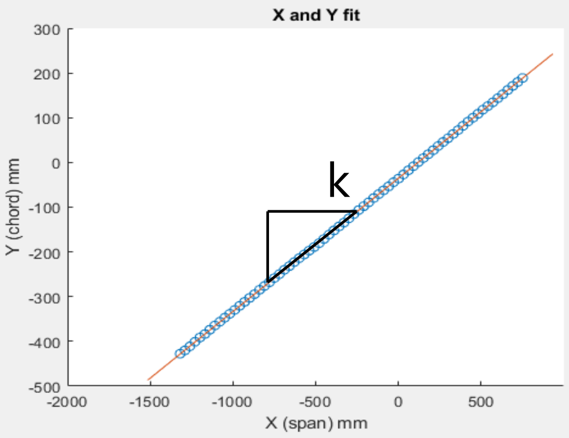


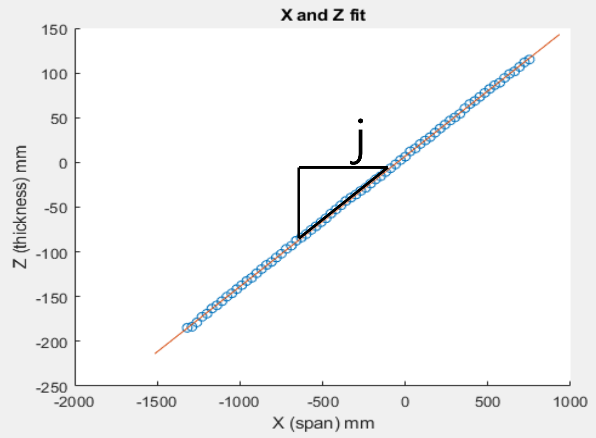
Leading-edge points are rotated back to blade coordinates

I – rotation about the span axis (x) – same as twist angle
J – rotation about the chord axis (y) – determined from X and Z fit
K – rotation about the thickness axis (z) – determined from X and Y fit

The orientation of the robot end effector is determined by the geometry of the blade



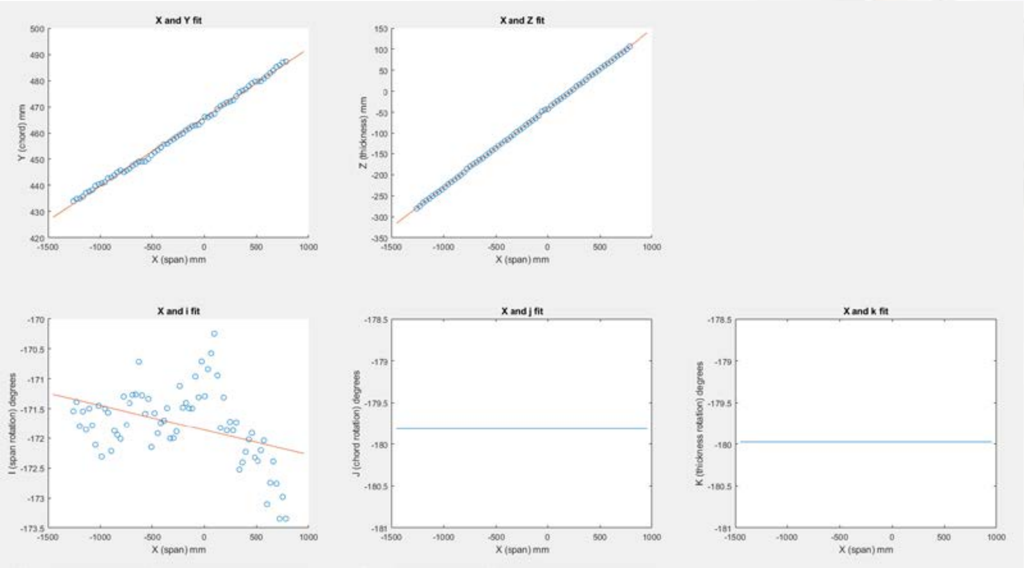




ACMA
American Composites Manufacturers Association

# Point Cloud Processing – Part 3

PROCESS

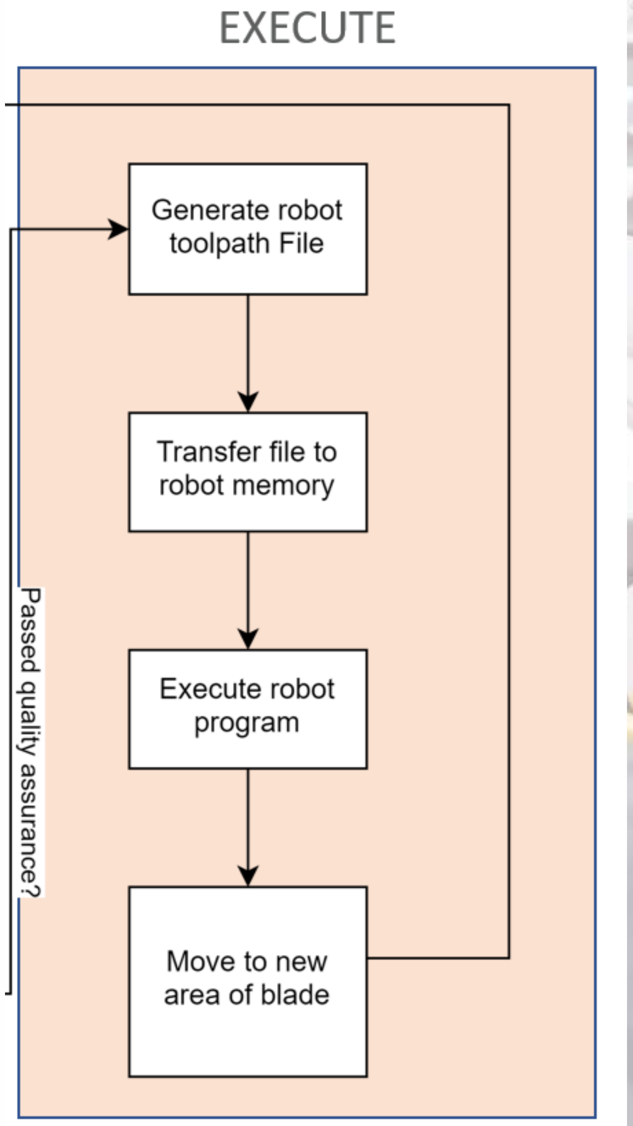
The toolpath quality is measured by fitting the data using a regression analysis. There should not be any large changes in position and orientation throughout the toolpath. First order or higher polynomials may be used.




The final toolpath is generated by writing a text file of leading-edge positions and orientations. These can be described with homogeneous transformations.

```
val(:,:,1) =

    1.0000   -0.0009   -0.0032   -1.4523
    0.0005    0.9884   -0.1520    0.4278
    0.0033    0.1519    0.9884   -0.3159
         0         0         0    1.0000


val(:,:,2) =

    1.0000   -0.0009   -0.0032   -1.4218
    0.0005    0.9884   -0.1517    0.4286
    0.0033    0.1517    0.9884   -0.3102
         0         0         0    1.0000


val(:,:,3) =

    1.0000   -0.0009   -0.0032   -1.3914
    0.0005    0.9885   -0.1515    0.4294
    0.0033    0.1515    0.9884   -0.3044
         0         0         0    1.0000
```

# System Integration



# When the Toolpath Generation Process Is Complete, the Program Can Be Uploaded to the Robot and Executed

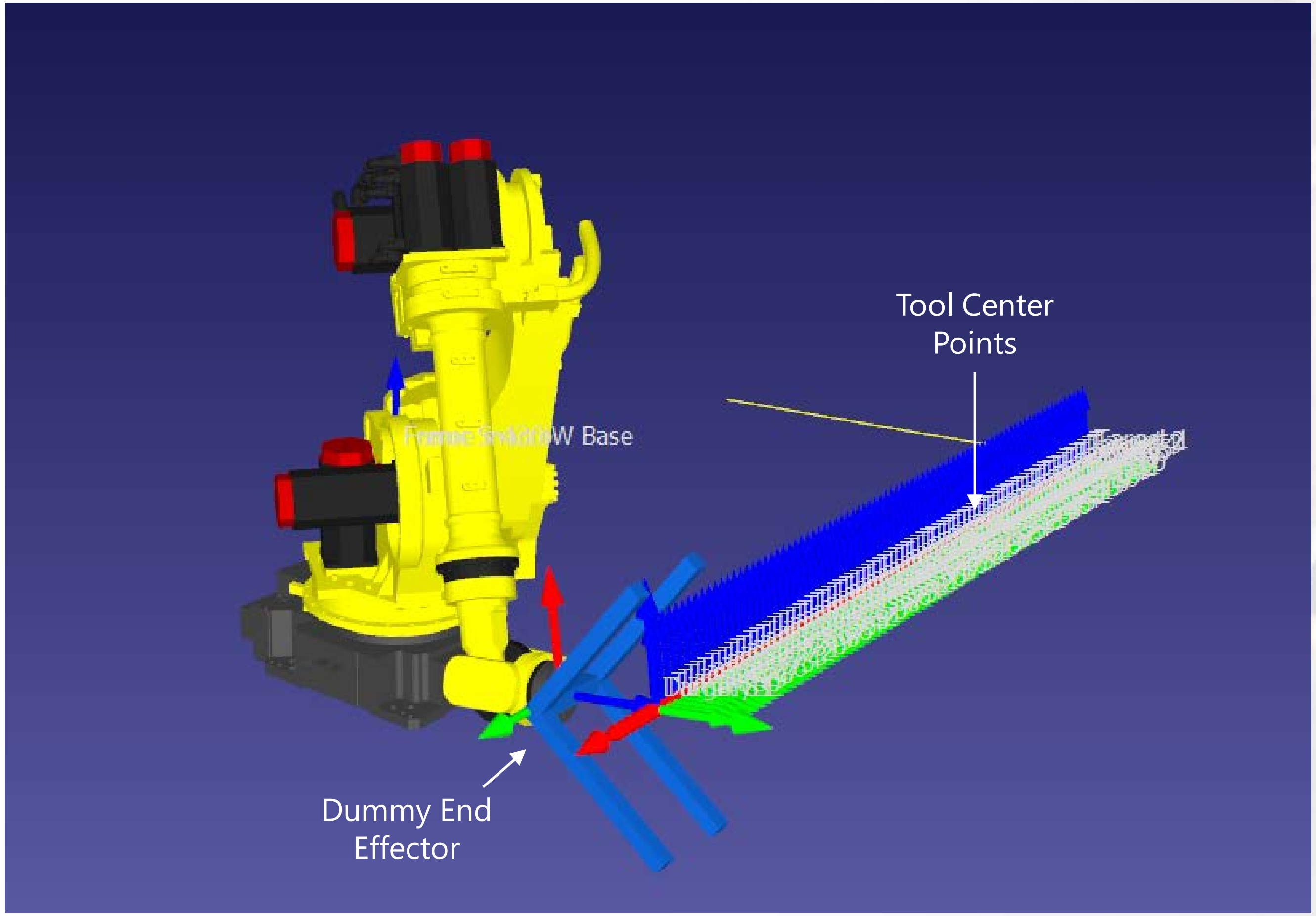
EXECUTE

- Final steps may include:
  - Including any offsets necessary for end effector design
  - Conducting toolpath quality assurance (by technician)
- This step is robot-dependent, meaning:
  - There are several ways to turn a list of tool center points into an executable robot file
  - One way to simulate the toolpath is with an offline robotics simulator like RoboDK.

# Robot Simulation of Toolpath in RoboDK

EXECUTE


Writing toolpath to RoboDK simulation software with MATLAB



# Future Work

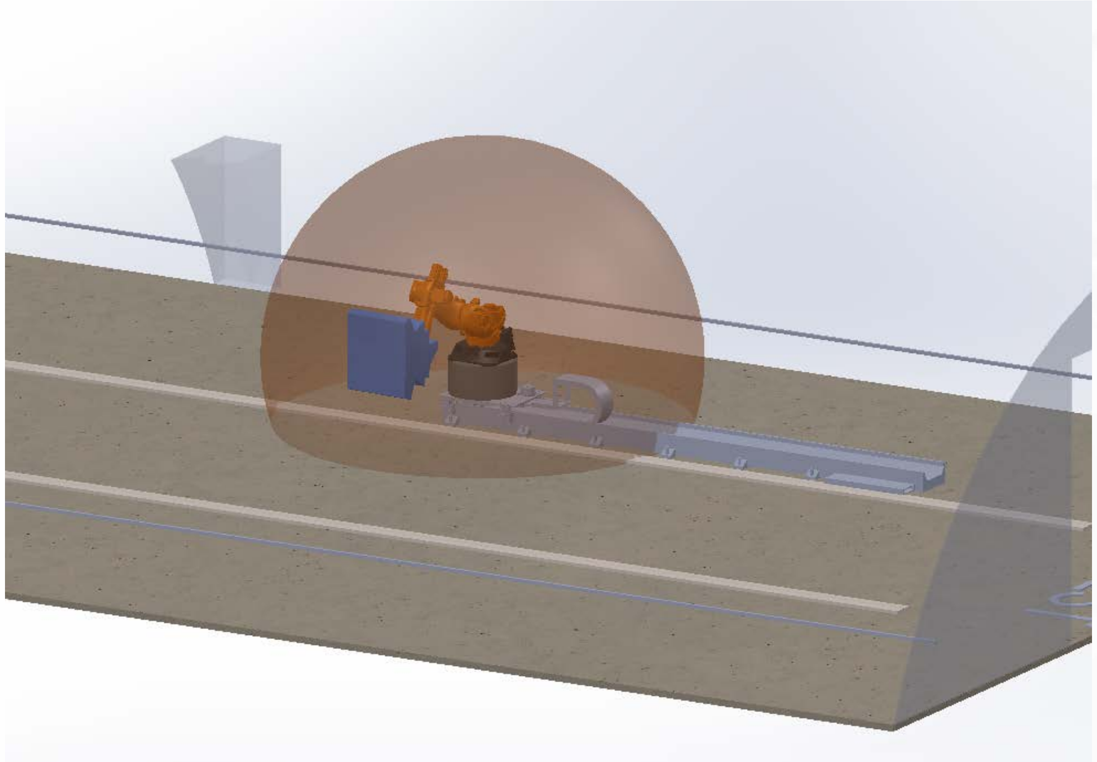
Continuing the development of a fully automated blade-finishing system

# NREL-GE-LM Wind: Grinding End Effector Research

- The toolpath just defines the 6-degree-of-freedom position of the leading edge, which requires the robot end effector to adapt to the varying profile of the airfoil.
- Novel end effectors are under development with project partners to reduce cycle time and maximize efficiency.

# NREL Robotic Research Platform

- The Kuka KR300 R2500 with a 6-m linear rail is in procurement and due for commissioning in September 2021.
- This system will further improve NREL's capabilities to design and test automated blade-finishing solutions.



# Opensource Code: Github

- All software will be migrated to NREL Github and made available to the public for free use (late 2021).
- All software will be converted to exclusively Python (open source).
- Third parties may edit code, suggest revisions, and add modules to improve and add capabilities.
- Reach out if you would like to access this code sooner.

# Thank you for attending!

# Questions


By Dennis Schroeder, NREL 41790

# Acknowledgements

NREL: David Snowberg, Derek Berry, Scott Lambert, Ryan Beach, Scott Hughes, David Barnes

GE: Arvind Rangarajan, Younkoo Jeong, Valerio Crovasce

JR Automation: For providing a loaner robot to aid in research

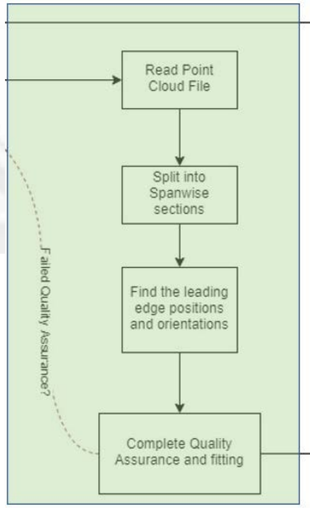
Contact Info:

Casey.Nichols@nrel.gov

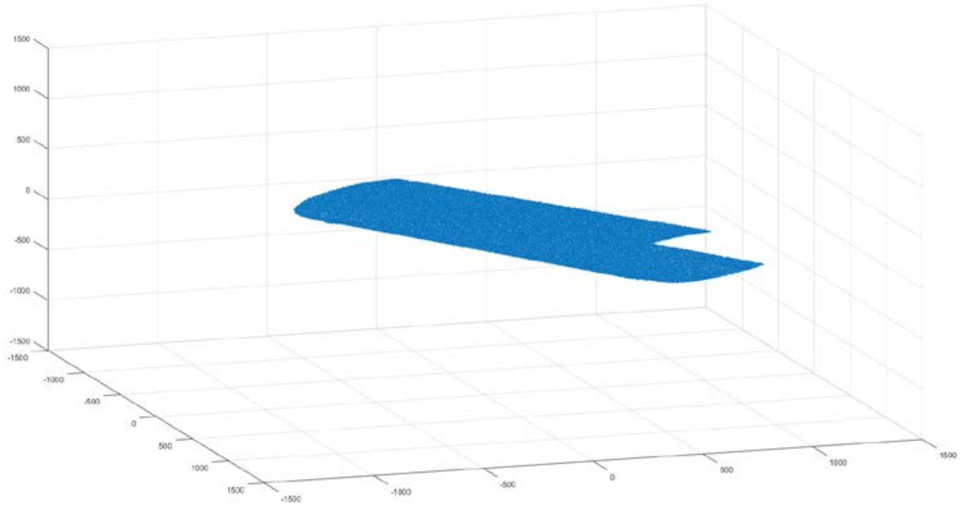
This work was authored [in part] by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Wind Energy Technologies Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

# Backup Slides

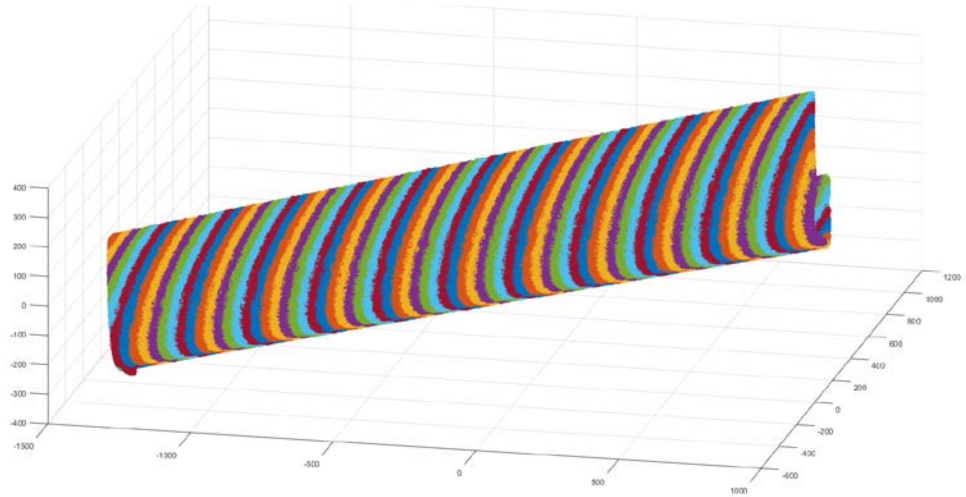
# Point Cloud Processing




Start with a filtered point cloud with

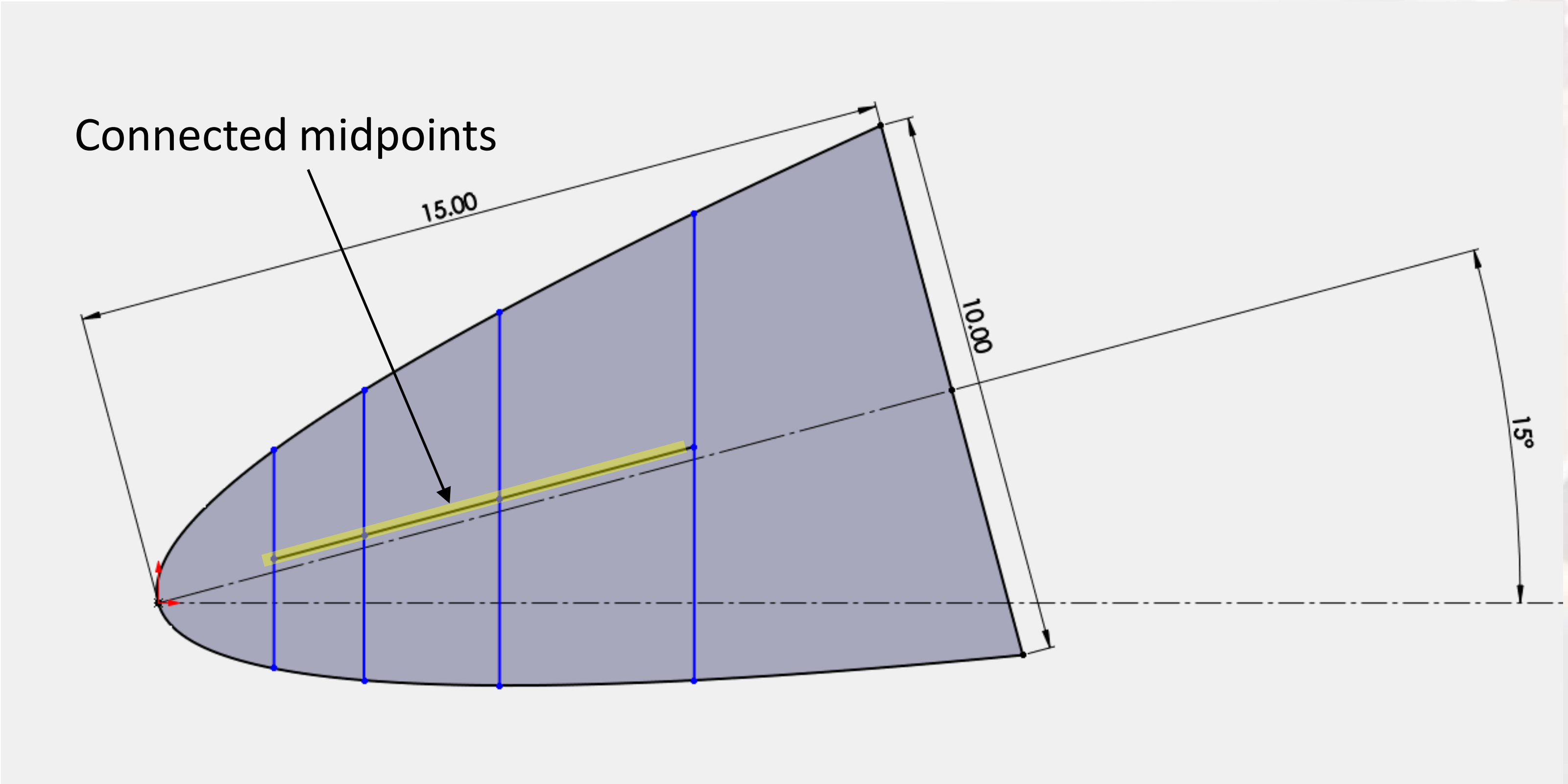


The point cloud will be broken down into spanwise sections to define partial airfoil geometry. The sections can be used to find the point of the leading edge while filtering 3D camera noise. This process is shown on the next slide.



# Finding Twist Angle

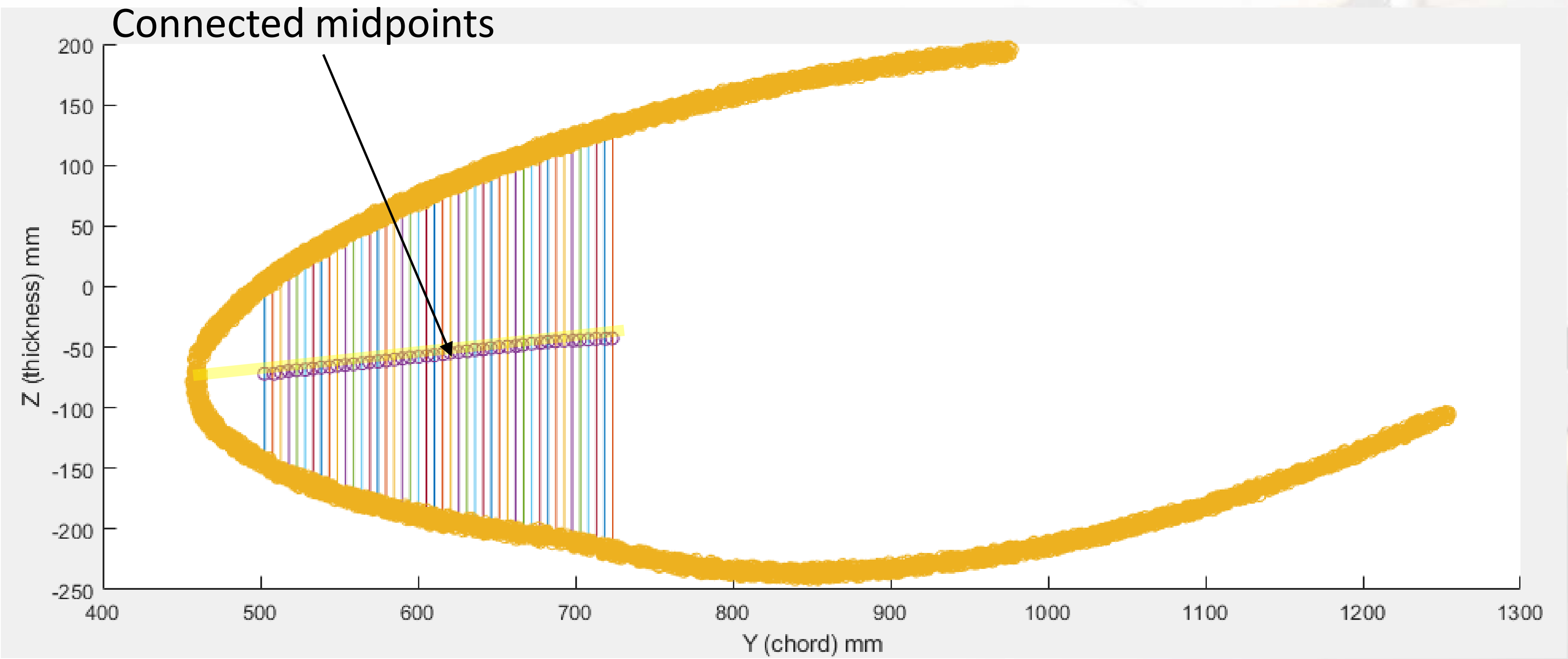
It is possible to find the twist angle by creating a series of vertical lines through the rotated parabola. Connecting the midpoints of the vertical lines creates a line that is parallel to the parabola centerline. The angle of this parallel line can be used to rotate the point cloud.



# Finding Twist Angle

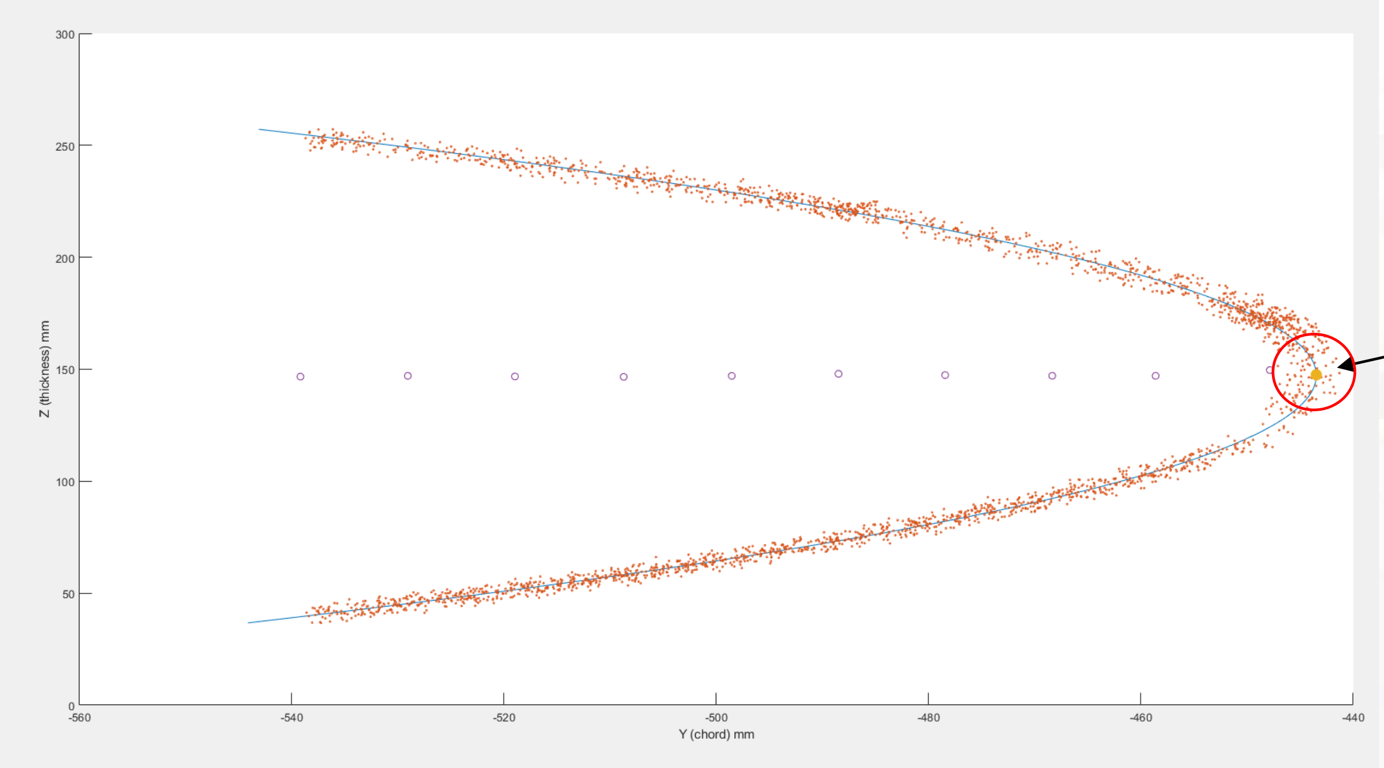
Implemented in MATLAB.

The parabola is then rotated by the angle determined by a linear fit of the midpoint line.



# Finding the Leading Edge with Parabola Polyfit

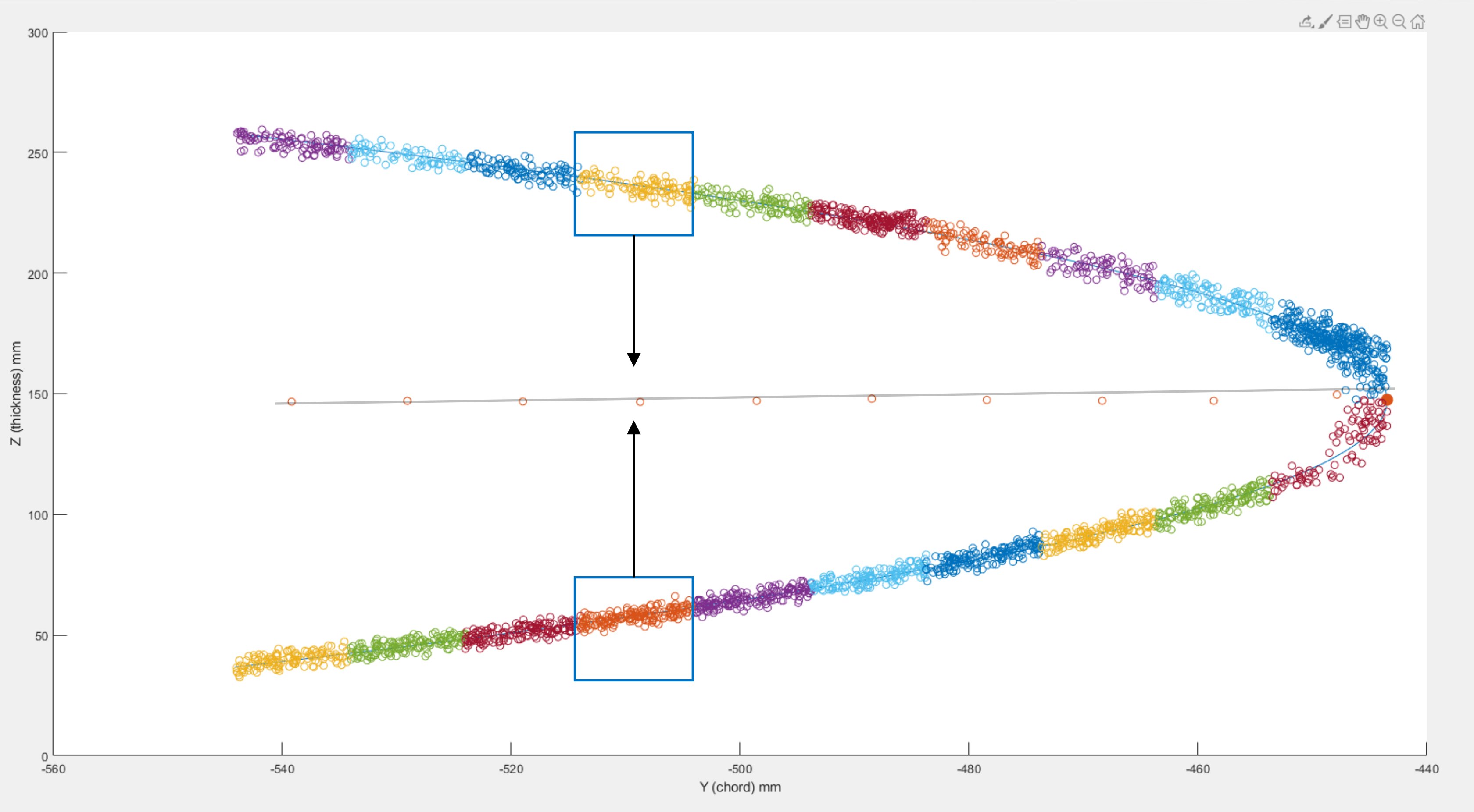
Polyfit will find the leading-edge point through the noise of the point cloud. Statistical methods also allow for the calculation of the root-mean-square (RMSE) for error detection. If the RMSE value is too high, an error has occurred.




Leading-edge point – vertex of parabola

# Second Twist Angle Check With Centerline

Colored sections were averaged to find the centerline. This method finds the centerline and is **<u>less influenced by point cloud density variations</u>**.

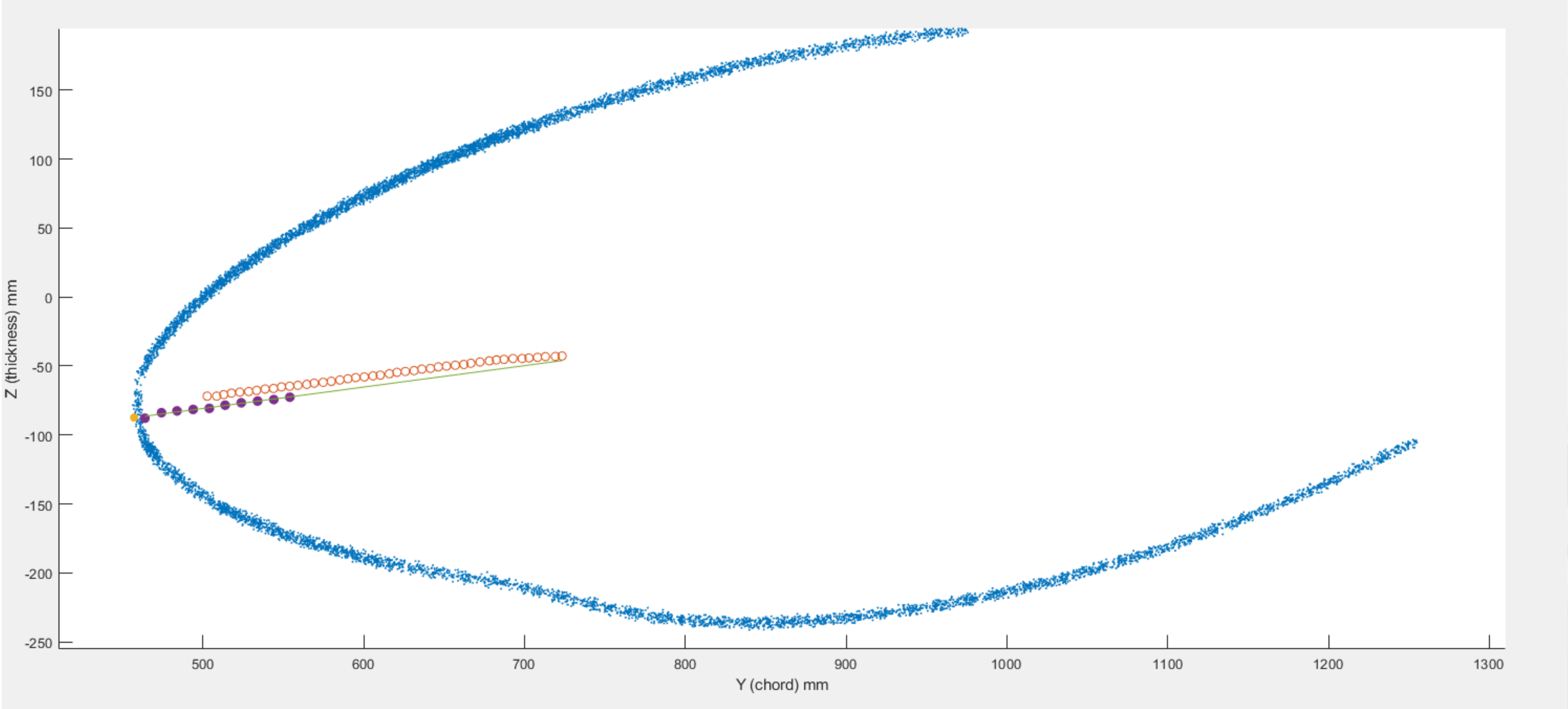


# Leading-Edge Point and Parabola Line Brought Back to Original Orientation

Orange dots – first twist angle determination with center points

Purple dots – Second twist angle determination from parabola

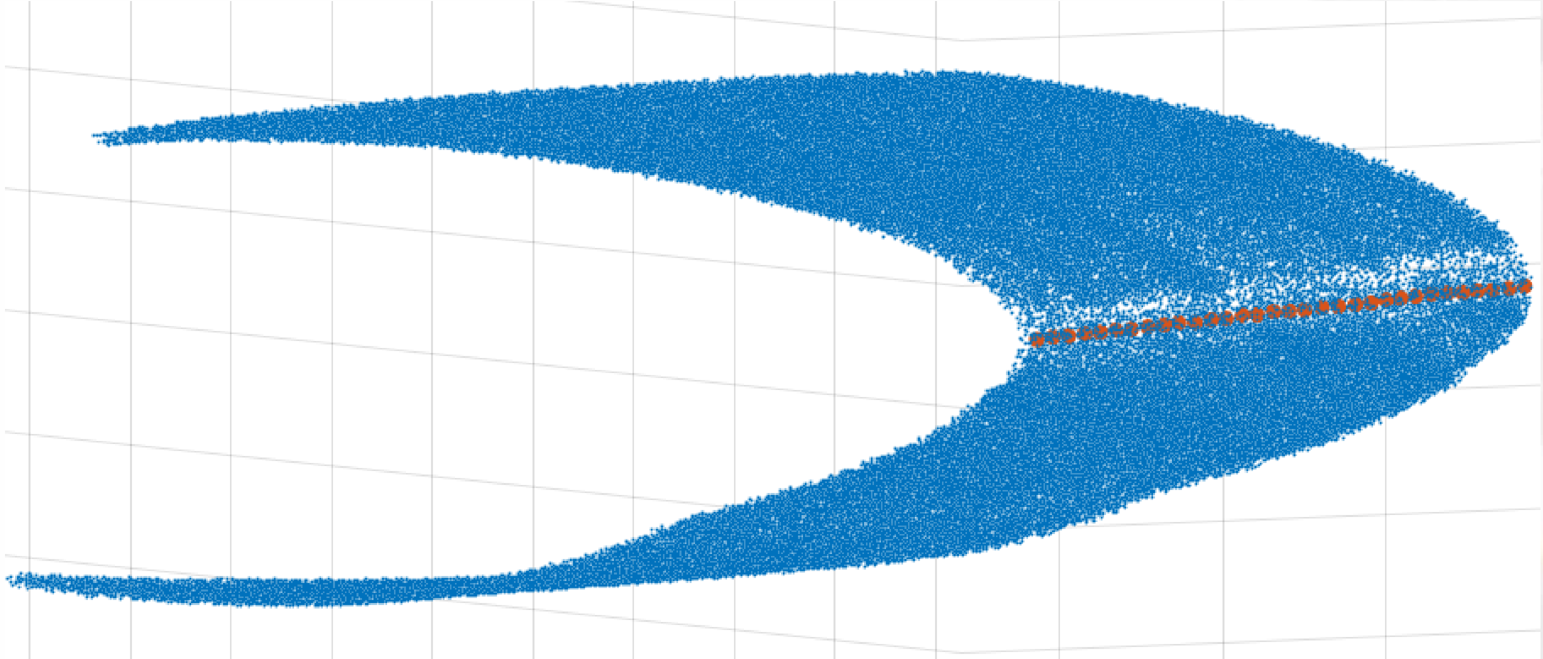
Green line – linear fit of blade section centerline



# This Process Is Repeated For All the Complete Span Sections

This establishes another set of data that can be fitted to improve accuracy and lower the impact of noise

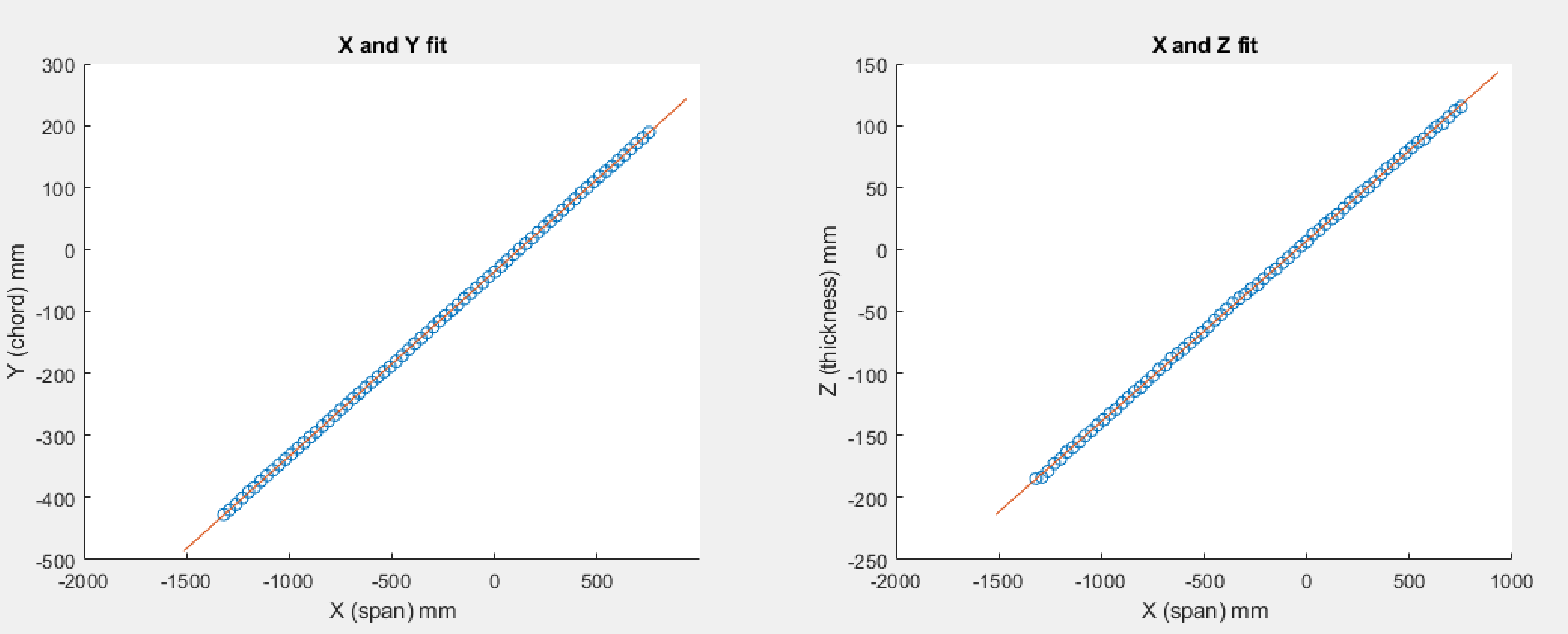
The data are fit with another polynomial fit. The order of the polynomial is decided by the RMSE value of the fit. First, second, and third order fits are attempted, and the lowest order is selected as long as it is under a RMSE threshold. This prevents overfitting.



# Fitting the Leading Edge in the Spanwise Direction

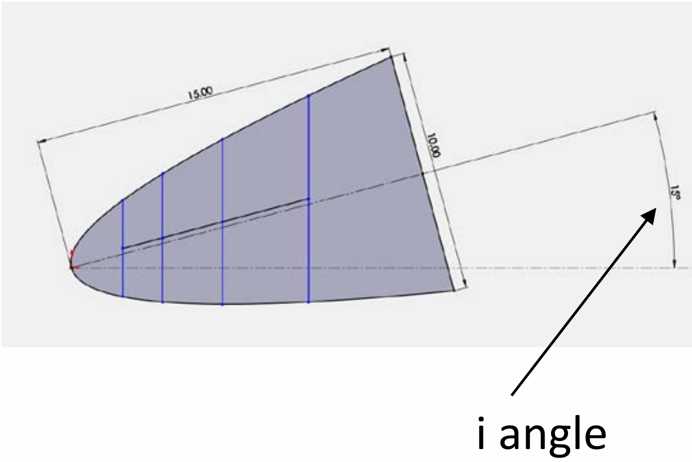
The appropriate fit for these data is a linear. Although, a higher-order polynomial fit will reduce the RMSE value, an order of 1 is used because it is below the threshold of an acceptable RMSE. Therefore, the data is not overfitted.

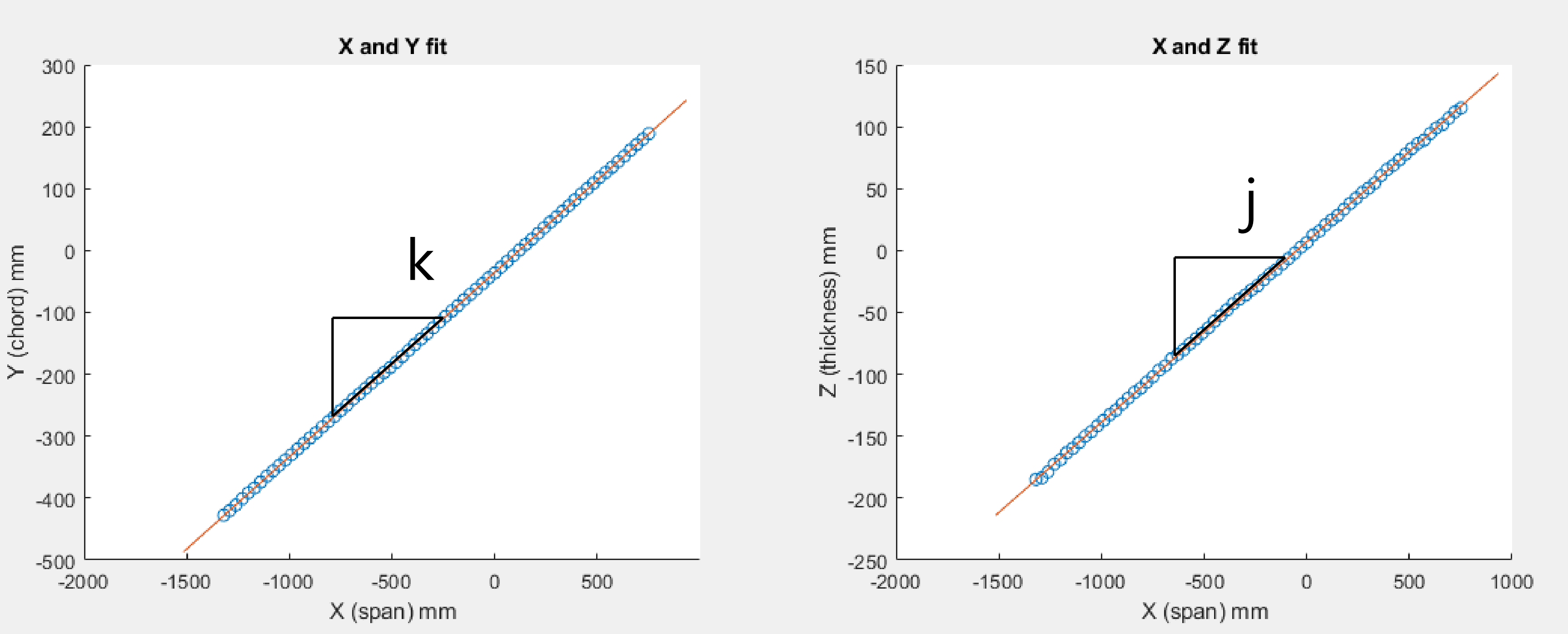
-blue dots – leading-edge points from parabola -orange line – leading-edge polynomial fit



# End Effector Orientation Determination

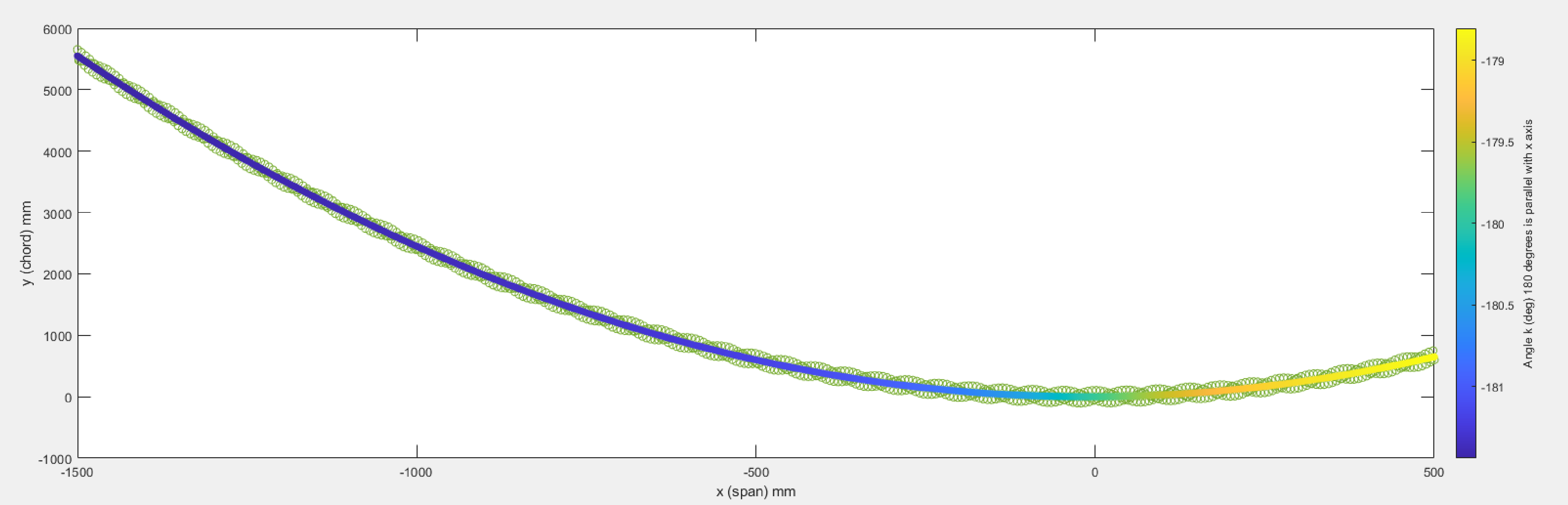
I – rotation about the span axis (x) – same a twist angle – determined from previous section
J – rotation about the chord axis (y) – determined from X and Z fit
K – rotation about the thickness axis (z) – determined from X and Y fit



# End Effector Orientation Determination

Method will work for higher-order fits based on the derivative of the polynomial function. Example shown below (not to scale). The color represents angle k.

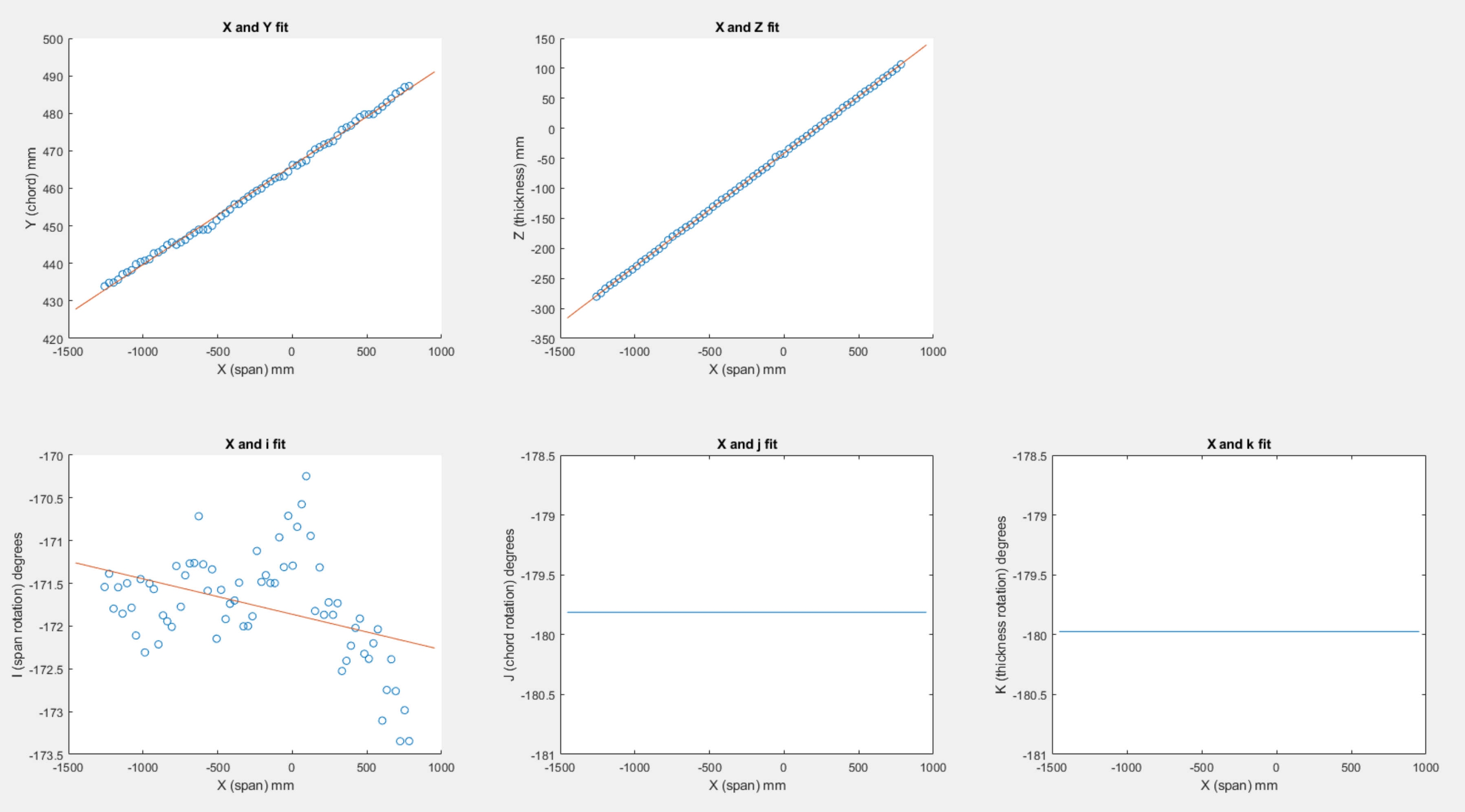


# Final Fit Quality

RMSE's respectively:

X/Y = 0.78 mm
X/Z = 1.08 mm
X/I = 0.5537 degrees

X/J and X/K depend on the position fits



# Homogenous Transformation Matrix Generation

A homogeneous transformation matrix is created in MATLAB and is written as a txt file for robot interpretation. Uses x, y, z, i, j, and k to create a matrix that can be used in inverse kinematics to program robot motors.

```
val(:,:,1) =

    1.0000   -0.0009   -0.0032   -1.4523
    0.0005    0.9884   -0.1520    0.4278
    0.0033    0.1519    0.9884   -0.3159
         0         0         0    1.0000


val(:,:,2) =

    1.0000   -0.0009   -0.0032   -1.4218
    0.0005    0.9884   -0.1517    0.4286
    0.0033    0.1517    0.9884   -0.3102
         0         0         0    1.0000


val(:,:,3) =

    1.0000   -0.0009   -0.0032   -1.3914
    0.0005    0.9885   -0.1515    0.4294
    0.0033    0.1515    0.9884   -0.3044
         0         0         0    1.0000
```

MATLAB

Matrix ${}^{i-1}_{i}T$ describes transformation from coordinates system „i" to coordinates system „i-1"

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i \cdot c\alpha_{i-1} & c\theta_i \cdot c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} \cdot d_i \\ s\theta_i \cdot s\alpha_{i-1} & c\theta_i \cdot s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} \cdot d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

In transformation matrix are used shortened symbols for trigonometric functions

$$ s\theta_i \rightarrow sin\theta_i \; oraz \; c\theta_i \rightarrow cos\theta_i $$

$$ s\theta_i \rightarrow sin\theta_i \; and \; c\theta_i \rightarrow cos\theta_i $$

Theory