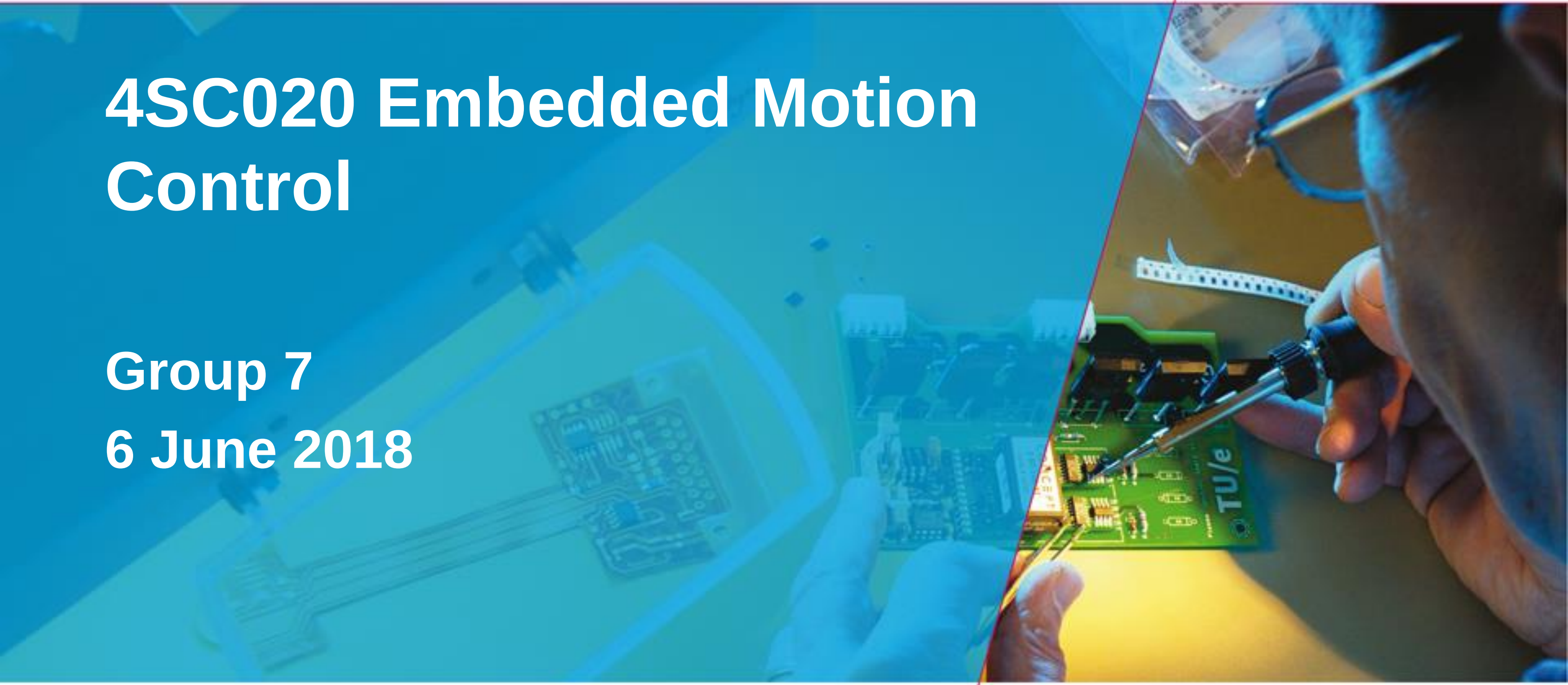
4SC020 Embedded Motion Control

Group 7

6 June 2018



Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Contents

1. Requirements
2. Proposed Approach
3. Function Implementation
   1. Mapping
   2. Localization

1. Requirements

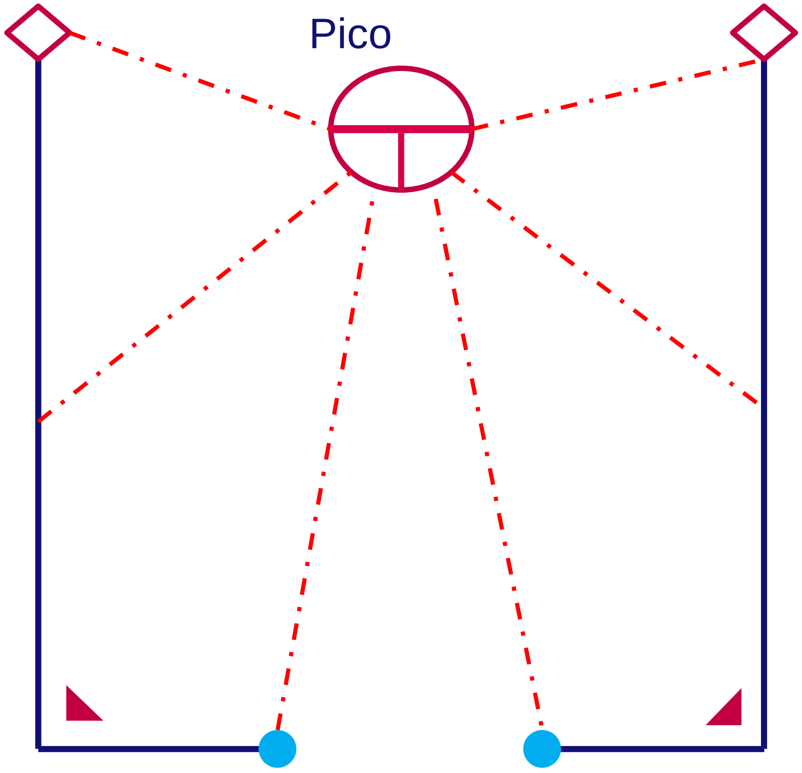
1. Requirements to functionality

1. Navigation
    1. Without hitting anything
    2. Position Pico
    3. Position ‘objects’ (exit, corners, wall)
    4. Navigation through hospital
2. Being able to map the whole hospital
3. Find the object
    1. Incorporate the hint into the model

2. Proposed Approach

2. Proposed Approach

- **Requirements 1.1 : Navigation without hitting anything**
- **Requirements 2 :   Being able to map the whole hospital**



Legend

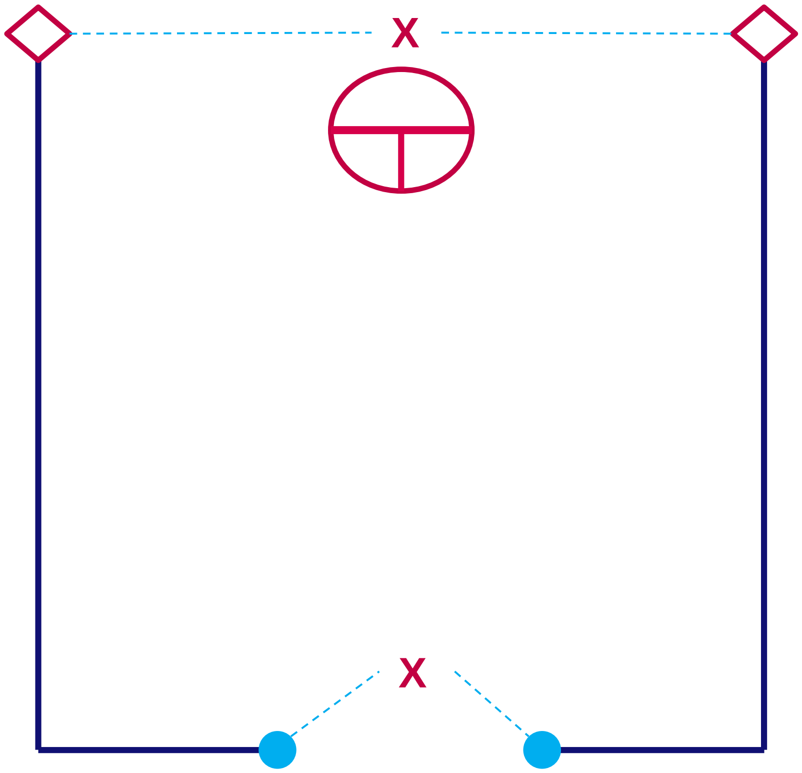
Intermediate point	◇
Exit corner	●
Corners	◣
LRF	-·-
Waypoint	X

2. Proposed Approach

- **Requirement 1.2 Position Pico**
  - **Positioning Pico only with odometry is too inaccurate**
  - **Narrow hallways (+- 0.5 m)**
  - **Corrections with laser data**

2. Proposed Approach

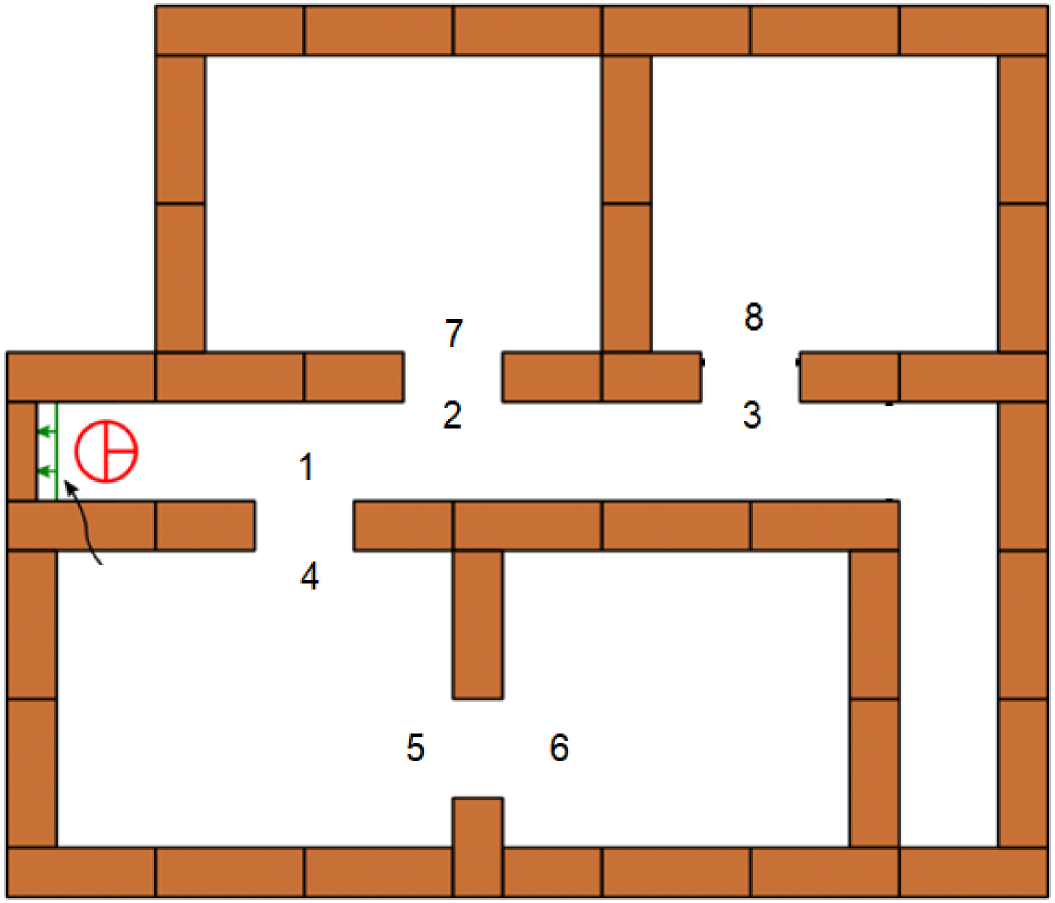
- **Requirement 1.4: Navigation through hospital**



Legend

Intermediate point	◇
Exit corner	●
Corners	◣
LRF	– · –
Waypoint	X

2. Proposed approach



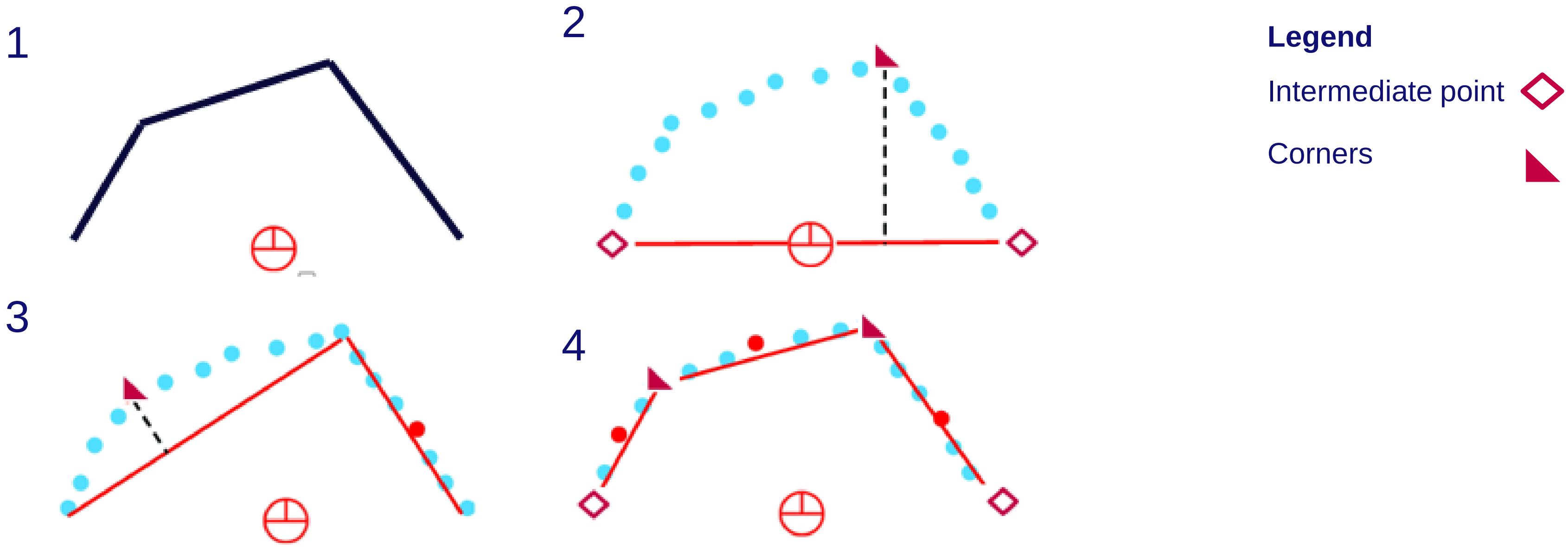
From: http://cstwiki.wtb.tue.nl/index.php?title=Embedded_Motion_Control

3. Function Implementation

- Discuss two functions:
  - Mapping
  - Localization of Pico
- Other functions, at wiki (next week)

4. Function Implementation I: Mapping

- **Split and merge algorithm**
- **Data beyond pi/2 –pi/2 will be thrown away**



From: https://www.researchgate.net/figure/Color-online-The-split-and-merge-and-line-extraction-algorithm-line-features-in-red_fig5_295099713

4. Function implementation II: Localization

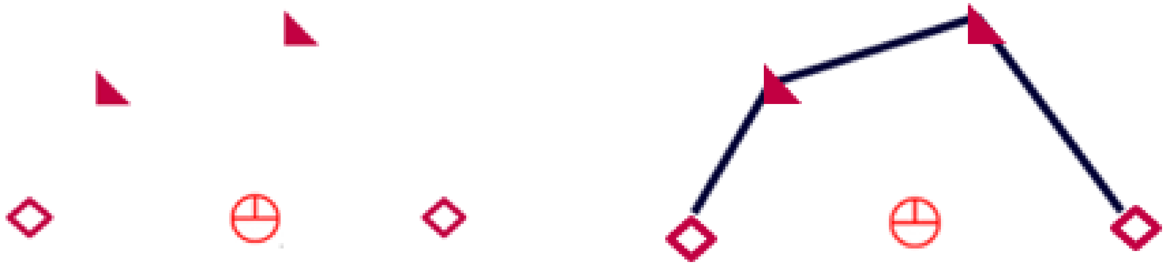
- **Odometry too inaccurate**
- **Exits as narrow as 0.5 m**
- **Use is made of laser data: compare LRF with walls**
- **Working principle:**

1. **Reference needed : markers mapped to OpenCV to obtain wall coordinates**
2. **Guess position, based on laser and Δodom**
3. **Compare LRF data in guessed position range (2) to wall coordinates (1)**
4. **Least square fit, best fit is position**

4. Function implementation II: Localization

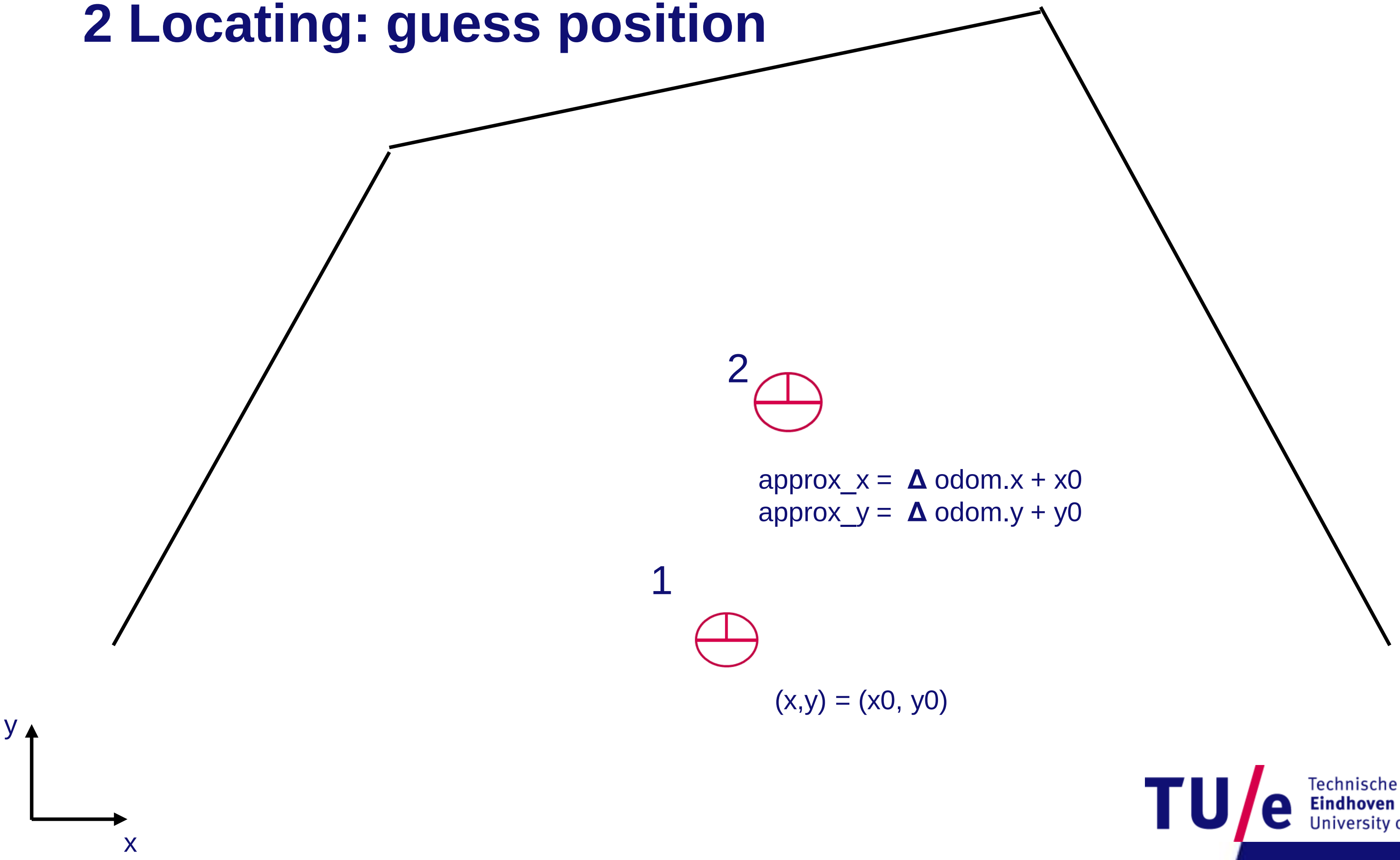
1 Mapping

- Mapping (OpenCV)
- Predefined image 3000x3000 px
- Corners located, draw lines between them



4. Function implementation II: Localization

2 Locating: guess position

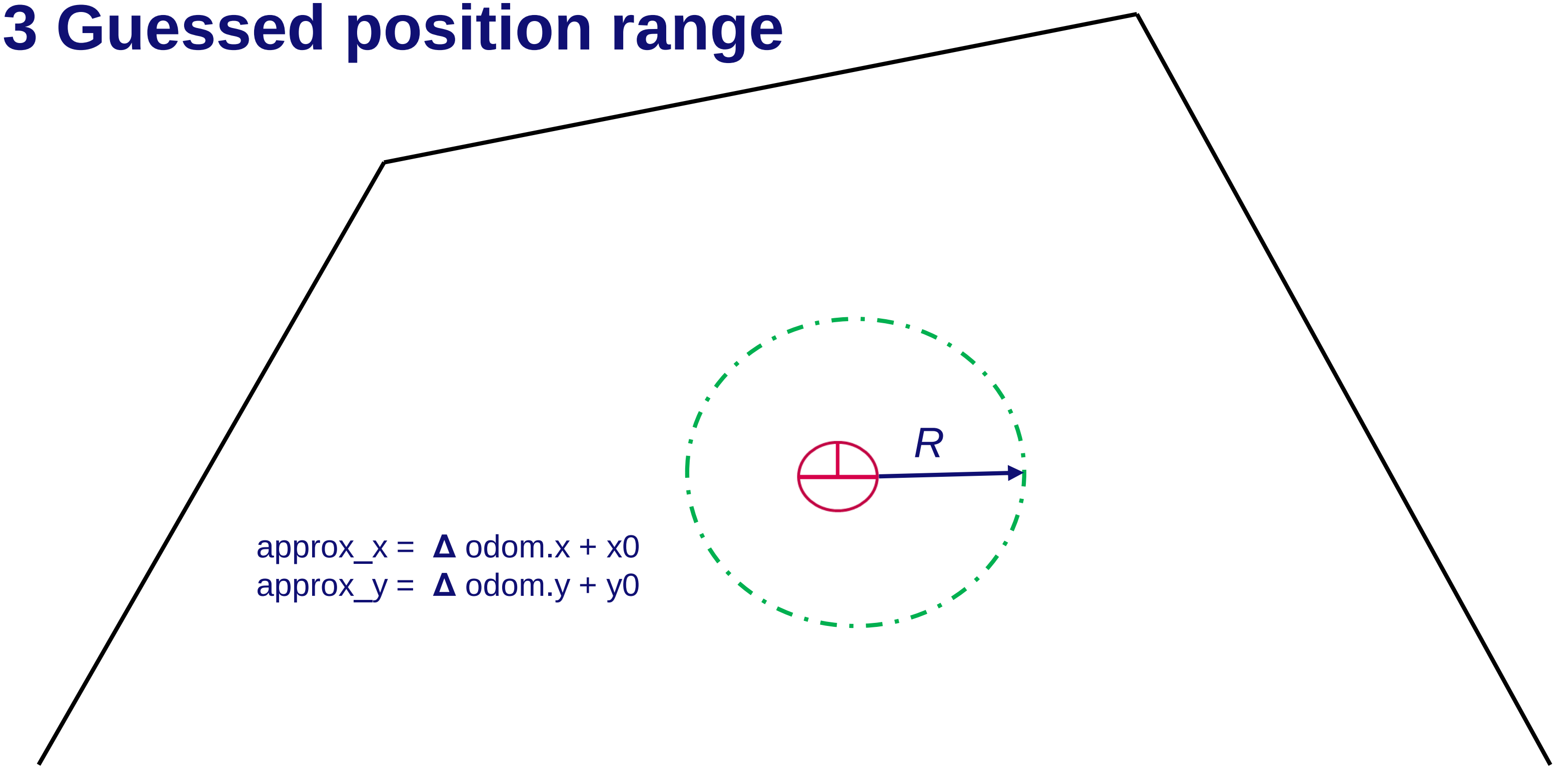


approx_x = Δ odom.x + x0
approx_y = Δ odom.y + y0

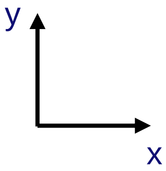
(x,y) = (x0, y0)

4. Function implementation II: Localization

3 Guessed position range



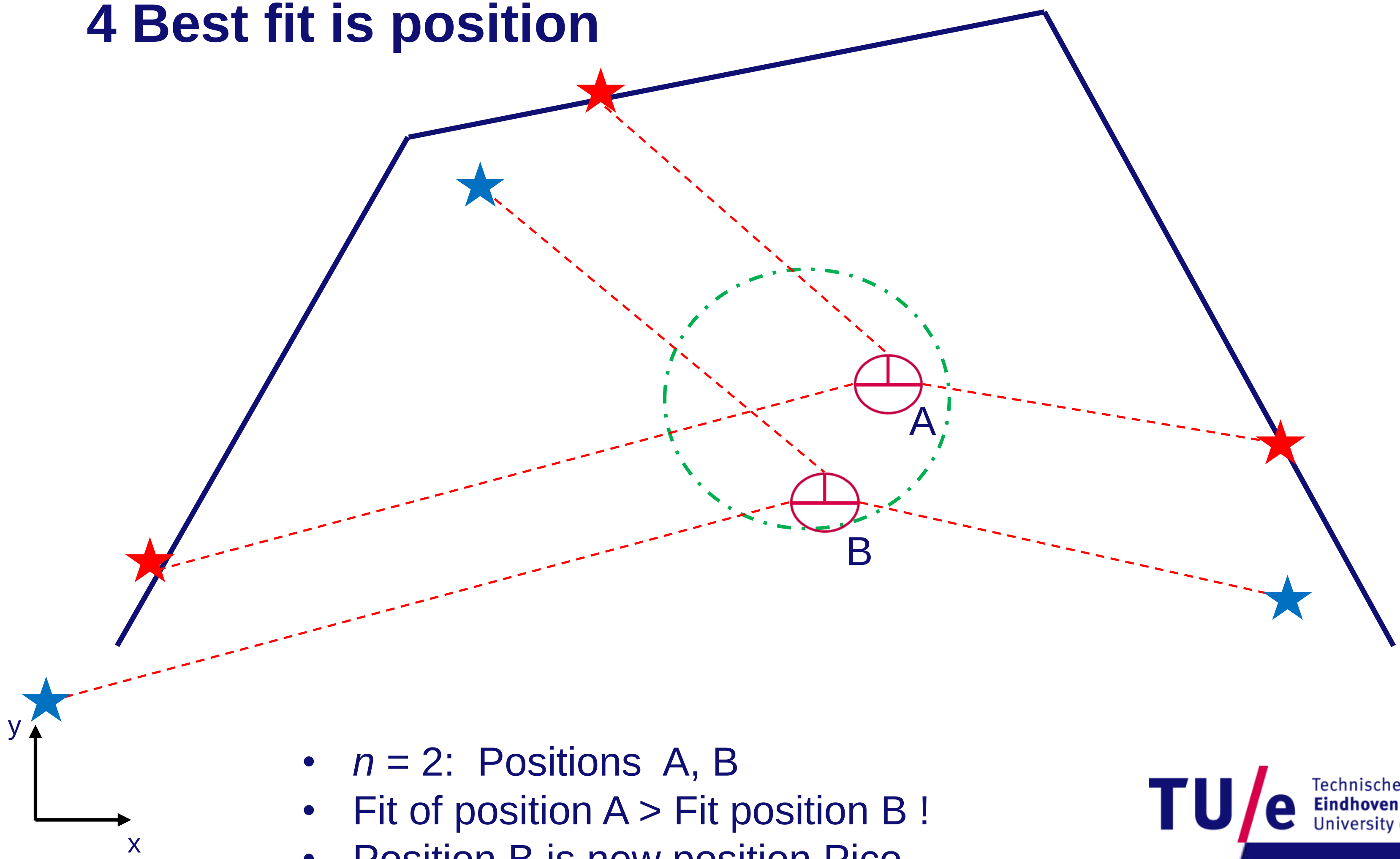
approx_x = Δ odom.x + x0
approx_y = Δ odom.y + y0



Assumption, Pico could in any of the *n* positions within range of *R*

4. Function implementation II: Localization

4 Best fit is position



- *n* = 2: Positions A, B
- Fit of position A > Fit position B !
- Position B is new position Pico

Summary

- Not hitting objects and mapping by wall coordinates of corners from split and merge algorithm
- Navigation with help of way point

Discussion

Appendix slides

2. Proposed approach

- **Requirement 3: Find object**
  - **Object is in farthest room, determine from waypoints which room it is**
  - **Determine room order for analyzing**
  - **Determine whether LRF data found within wall coordinates boundary**