

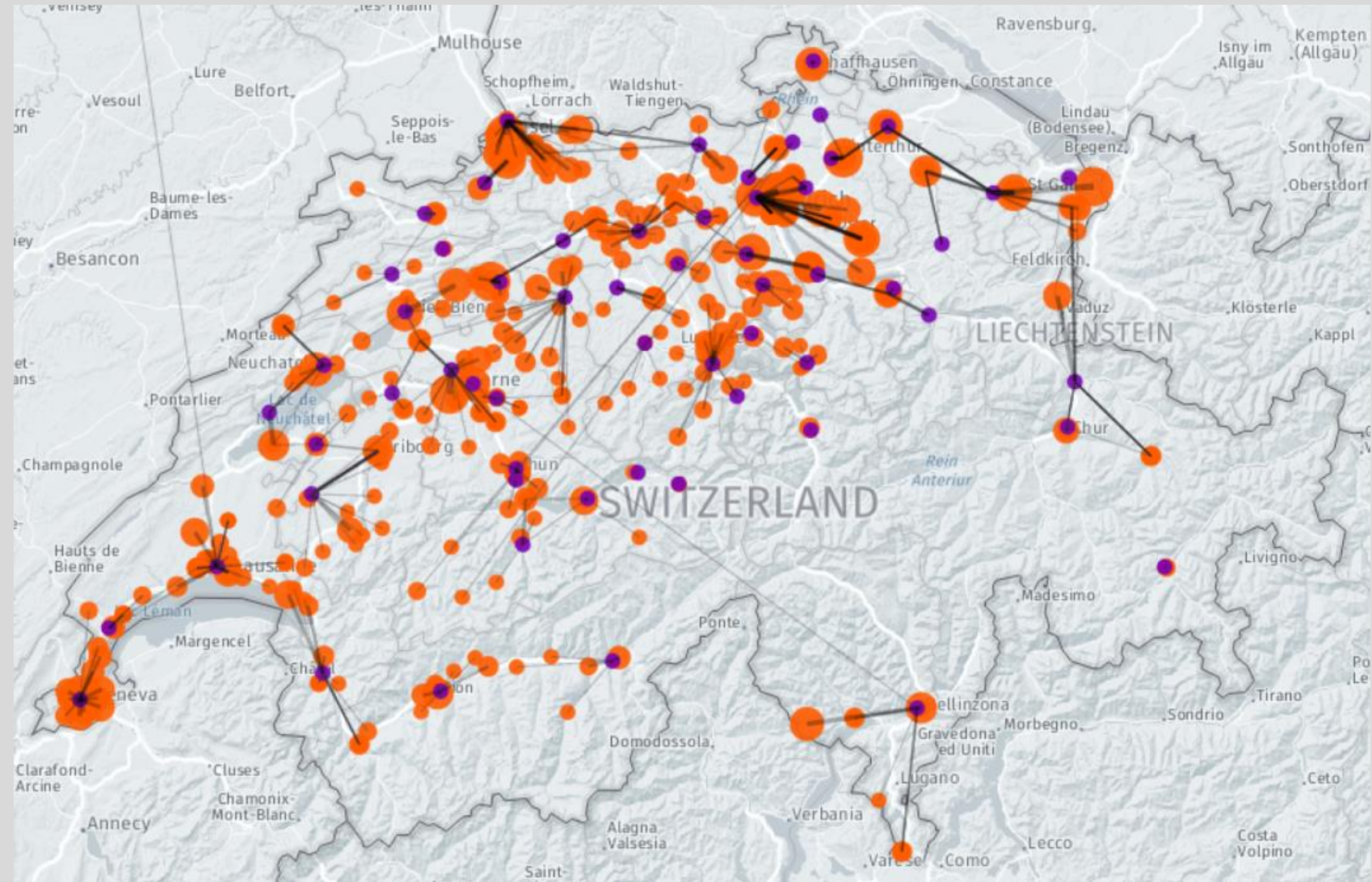
Accelerating Deployment of Autonomous Delivery Robots using Model Based Design

Dr. Erik Wilhelm
KYBURZ Switzerland

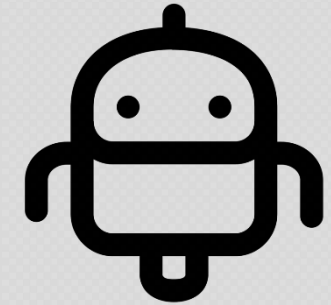


02.06.2020

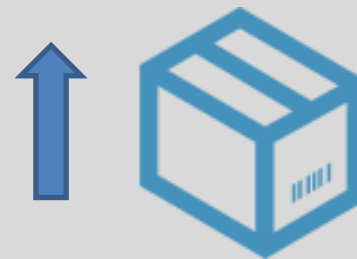
A well-established brand



Why Autonomous Delivery?



- Must be:
 - Cheaper
 - Faster
 - More reliable
 - ... More personal?



Four functional prototypes in 24 Months



- Mobile depot box (eT1 / eT2)
- Sensors
 - 2D Lidar
 - Ultrasonic
 - 360 camera
 - GPS
 - Bump-stop



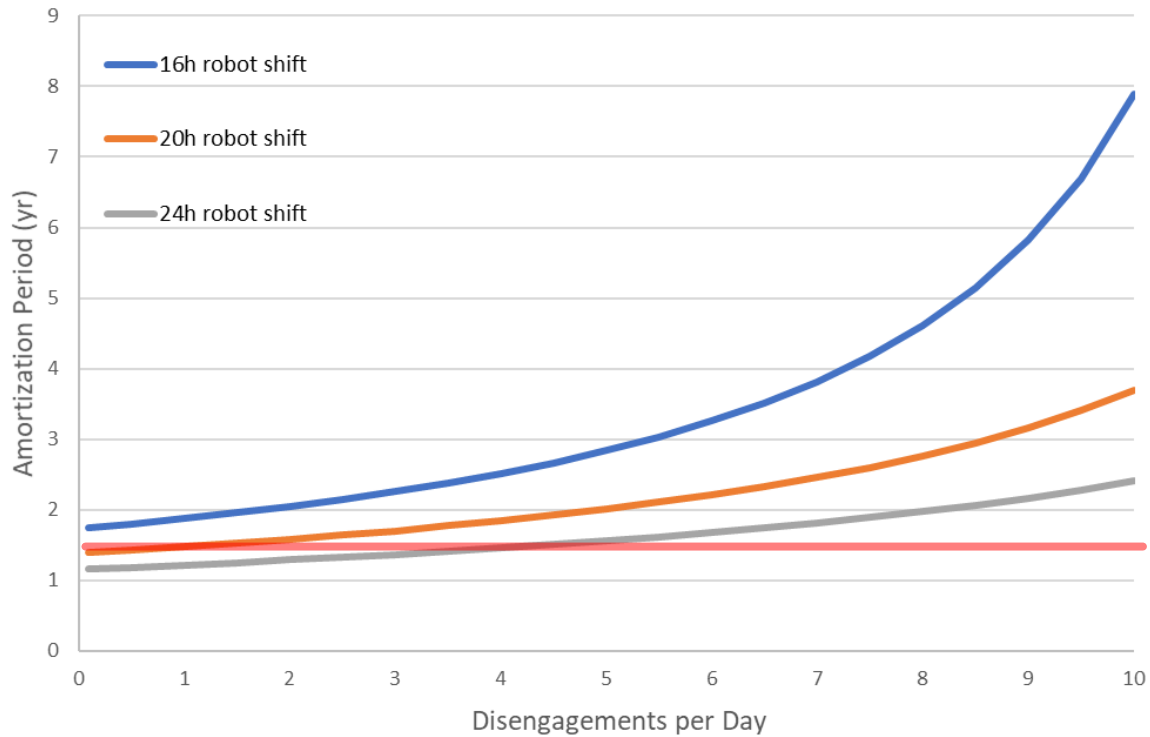
- Autonomous delivery agent (eT3)
- Sensors
 - 3D Lidar
 - Ultrasonic
 - Infrared
 - INS
 - Bump-stop



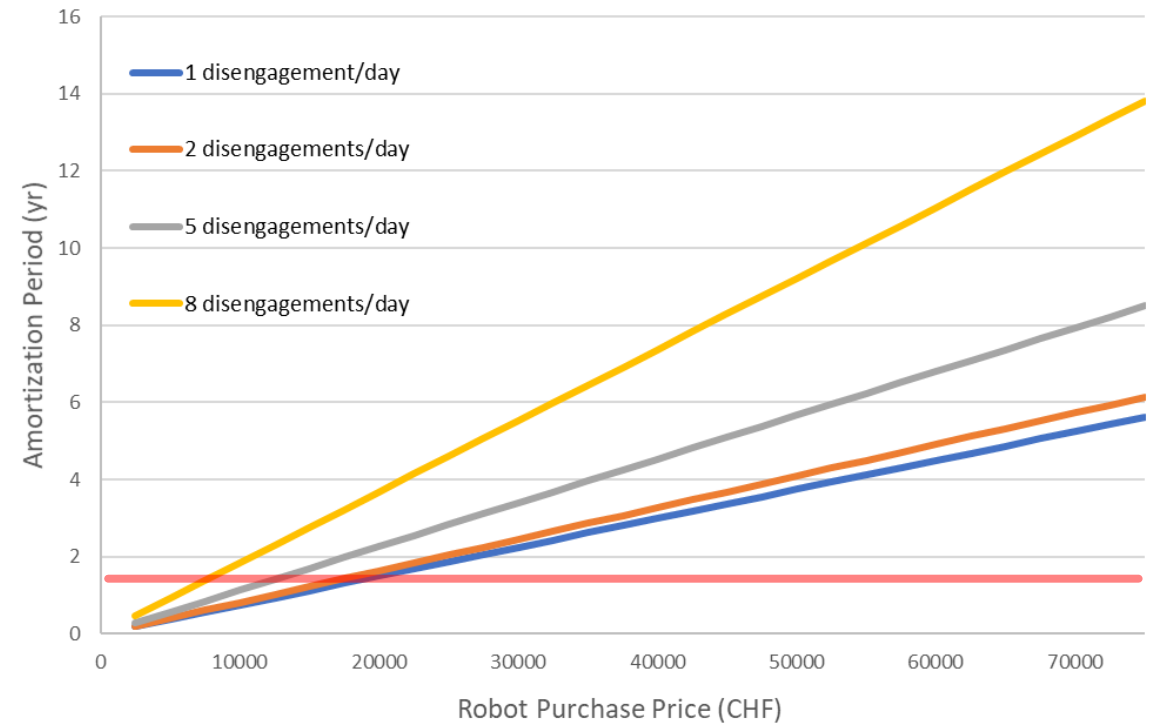
- Flexible delivery system (eT4)
- Sensors
 - 3D Lidar (2x)
 - Ultrasonic (12x)
 - Infrared (12x)
 - Radar
 - GPS (INS)
 - 360 Cameras (localization)
 - 360 Cameras (comprehension)
 - Time-of-flight camera
 - Bump-stop

Economics of Delivery Robots

Cost of Human Intervention



Sensor Cost



- Robots should be operated for long shifts without disengagement with cheap sensors

eT4: Autonomous delivery platform



Autonomous System Design Challenges

High availability



Image: ABC news

Approved safety

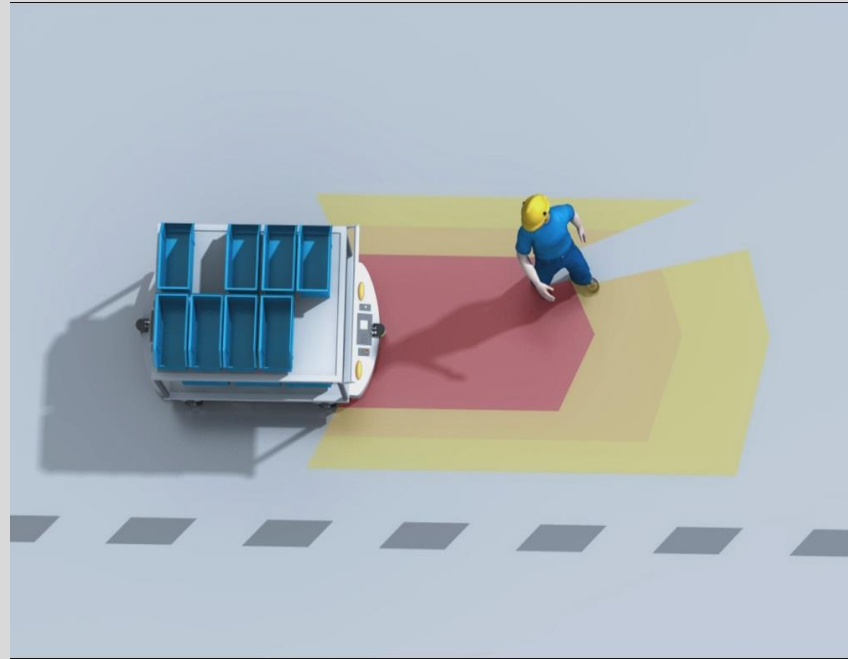


Image: sick.com

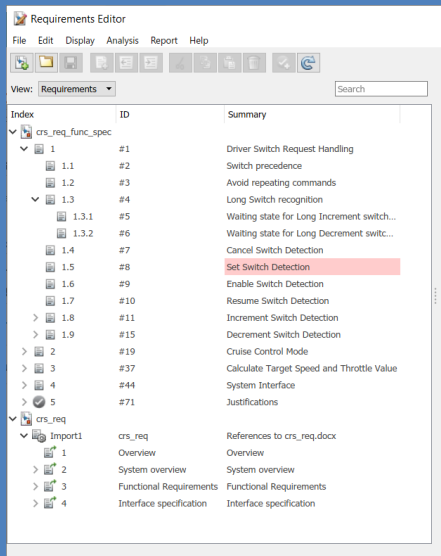
Test coverage



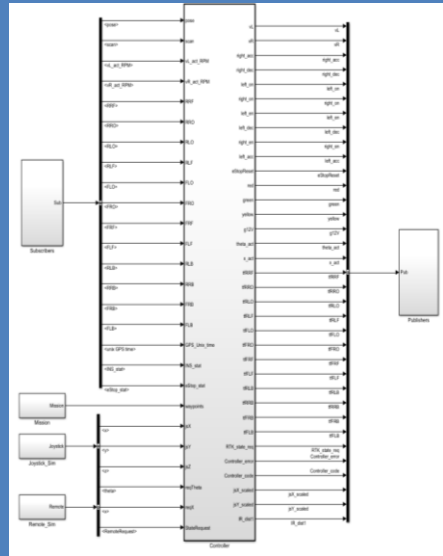
Image: youtube.com

Safety-centered workflow

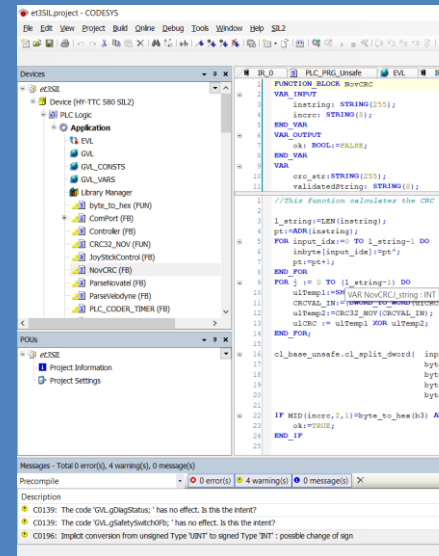
Specifications



Model Based Design



Code Generation



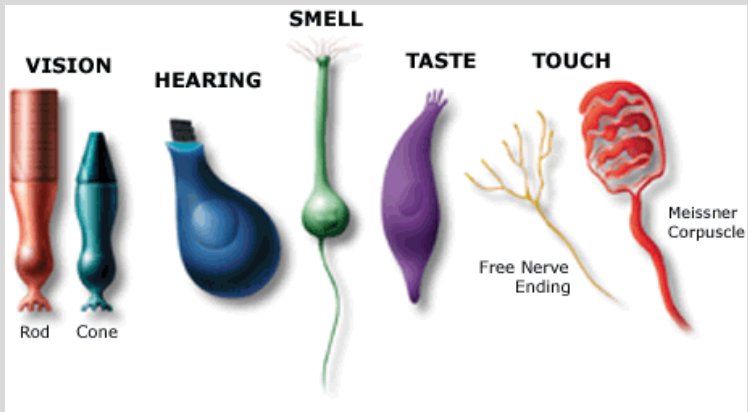
Compilation



- This workflow allows SIL2 certifiable code to be generated using model-based design
- Review and testing occurs within each phase and before each release

Incumbent

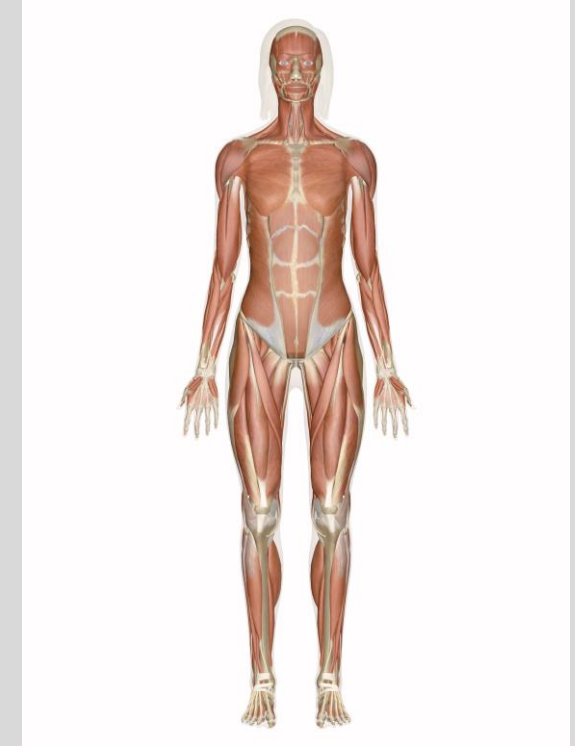
Sense



Infer



Act



100 – 200 ms

Challenger

Sense



40 ms



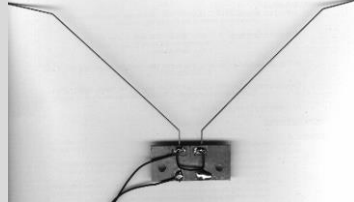
20 ms



300 ms



50 - 600 ms



20 ms

Infer



1 ms



300 ms

Act





20- 40 ms

650 - 820 ms

Does it have to be head to head?



Redundancy for sensor systems

Localization	Day	Night	Precipitation	Fog	Tall structures	Tranparent obstacles	Diffuse obstacles
INS (GPS)	✓	✓	✓	✓	!		
Optical	✓	✗	✗	!	✓		
Pointcloud	✓	✓	!	✗	✓	-	-
Obstacle Avoidance							
LiDAR	✓	✓	!	✗	-	✗	!
Optical	✓	✗	✗	!	-	✓	✓
Radar	✓	✓	!	✓	-	✓	✓
Ultrasonic	✓	✓	✓	✓	-	✓	!
Time-of-Flight	!	✓	!	✗	-	!	✗
Infrared	✗	✓	!	✗	-	✗	✗
Bump Stop	✓	✓	✓	✓	-	✓	✓

Integrated code-level functional specification

Requirements Editor

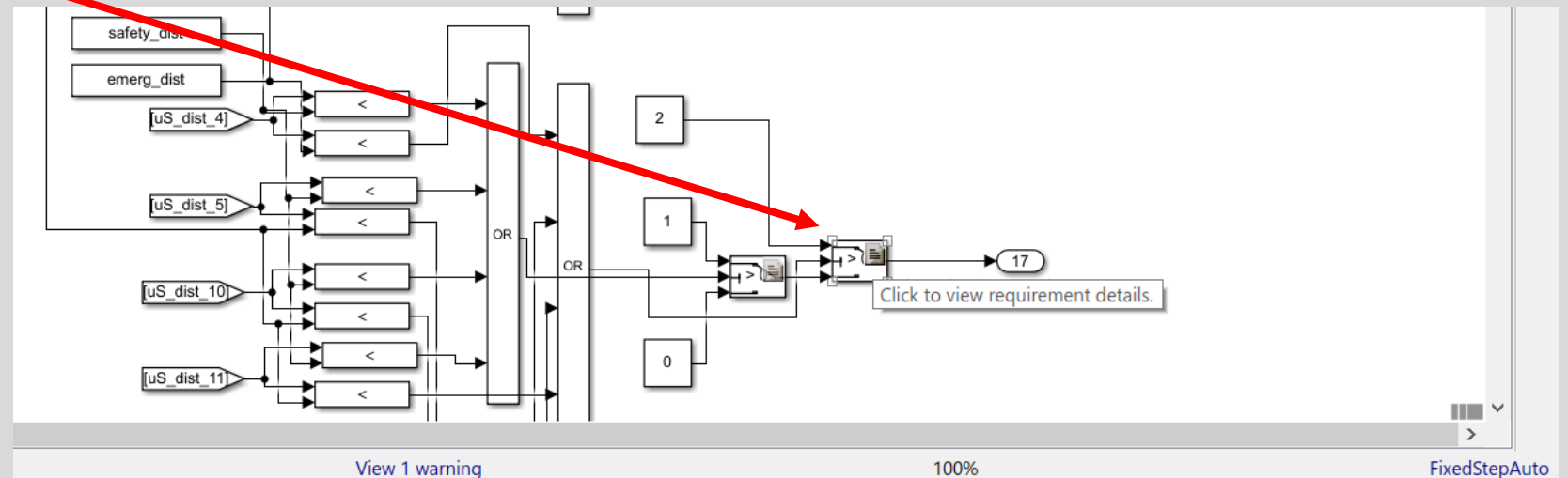
File Edit Display Analysis Report Help

View: Requirements

Index	ID	Summary
1	#1	The ultrasonic distance sensor error conditions should lead to an error message
2	#2	The ultrasonic distance sensor too close signal should be interpreted as an obstacle in the way
3	#3	The ultrasonic distance sensors will provide a safety zone warning
4	#4	The ultrasonic distance sensors will provide an emergency zone warning
5	#5	The sonar safety and emergency zones will provide a safety zone warning
6	#6	The sonar safety and emergency zones will be discriminated for rotation in both directions
7	#11	The lift controller will prevent hysteresis at the end points
8	#12	Obstacles result in slowing and stopping of vehicle
9	#13	Obstacle detection modes should be able to be selected by the CAN bus

Properties

Filepath: C:\Users\ew\Documents\GitHub\eT4_control\Simulink\data\eT4_req.slreqx
Revision: 9
Created by: ew
Created on: 20-Aug-2019 11:47:33
Modified by: ew
Modified on: 02-Sep-2019 11:53:32
Description:



Corner Cases

Field of Study: Hazard and Risk Analysis eT4																		
Item Definition - Functional Description			Hazard and Risk Analysis			ASIL Determination						Safety Goal		Allocation				
Operating Mode	ID	Function	Malfunction	Situation	Hazard	Severity	S-Comment	Exposure	E-comment	Controllability	C-Comment	ASIL	Safety Goal	Safe State	ID	Requirement	Element	
Driving Mode	F.01	Moving forward	Unintended moving forward	Downhill slope / Curvy Road	Crushing, impact with humans/kids	S3	Life-threatening injuries (survival uncertain),fatal injuries	E4	High probability	C3	Difficult control uncontrollable	D	SG.01	Unintended moving shall be prevented	Stopped	FSR.01	Motion command shall be provided correctly	VCU
																FSR.02	Freewheeling motor shall be prevented	Motor controller Motor
																FSR.03	Emergency stop signal shall be transmitted correctly	PDU
			FSR.04	Motion command shall be properly interpreted	Motor controller Motor													
			FSR.01	Motion command shall be provided correctly	VCU													
			FSR.03	Emergency stop signal shall be transmitted correctly	PDU													
Not moving forward	Train crossing / Crossover	Impact with others rolling part ejected	S3	Life-threatening injuries (survival uncertain),fatal injuries	E2	High probability	C2	Difficult control uncontrollable	A	SG.02	Unintended stop shall be prevented	Safe movement operation	FSR.01	Motion command shall be provided correctly	VCU			
													FSR.03	Emergency stop signal shall be transmitted correctly	PDU			
													FSR.05	Power shall be normally available in driving mode	PDU System (Power provider)			
FSR.06	Traction shall be available in driving mode	Motor controller Motor																
FSR.01	Motion command shall be provided correctly	VCU																
FSR.02	Emergency stop signal shall be transmitted correctly	PDU																
Incorrect moving	Falling down the stairs / Pedestrian area	Crushing, impact with humans/kids	S3	Life-threatening injuries (survival uncertain),fatal injuries	E3	Medium probability	C3	Difficult control uncontrollable	C	SG.03	Incorrect movement shall be prevented	Stopped	FSR.01	Motion command shall be provided correctly	VCU			



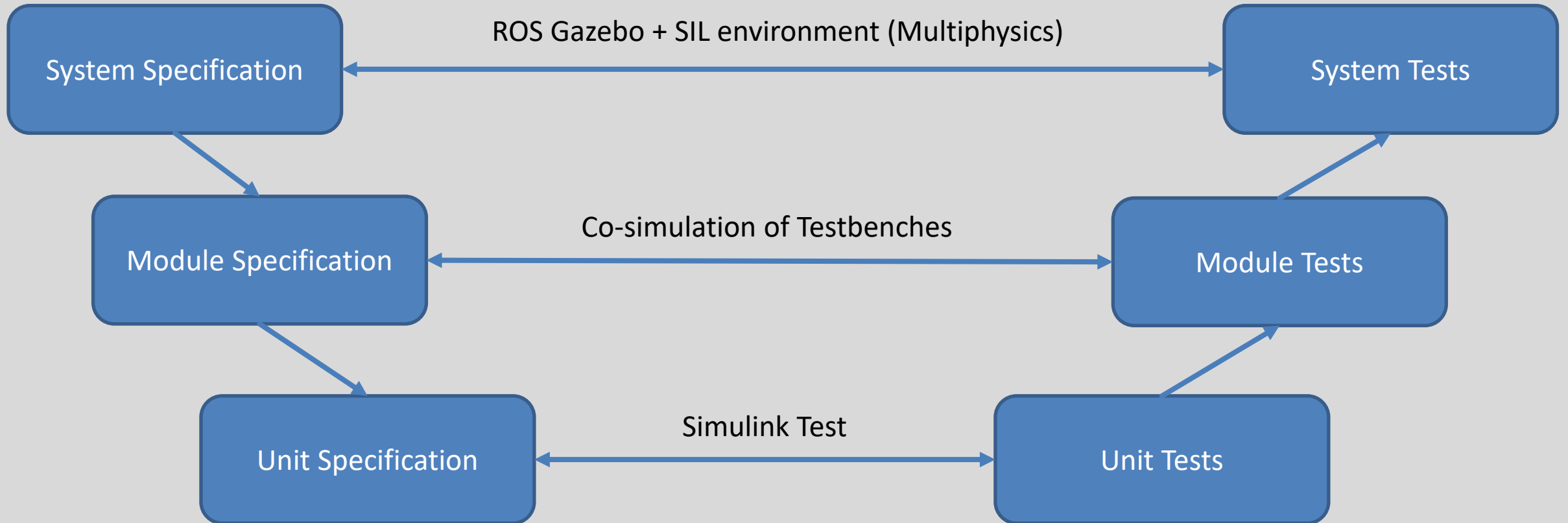
Image: drivingtests.co.nz



Image: arstechnica.com



Safety Solution



- Kyburz toolchain uses layered verification techniques and model-based design
- All requirements are easily documented for traceability

Unit Testing

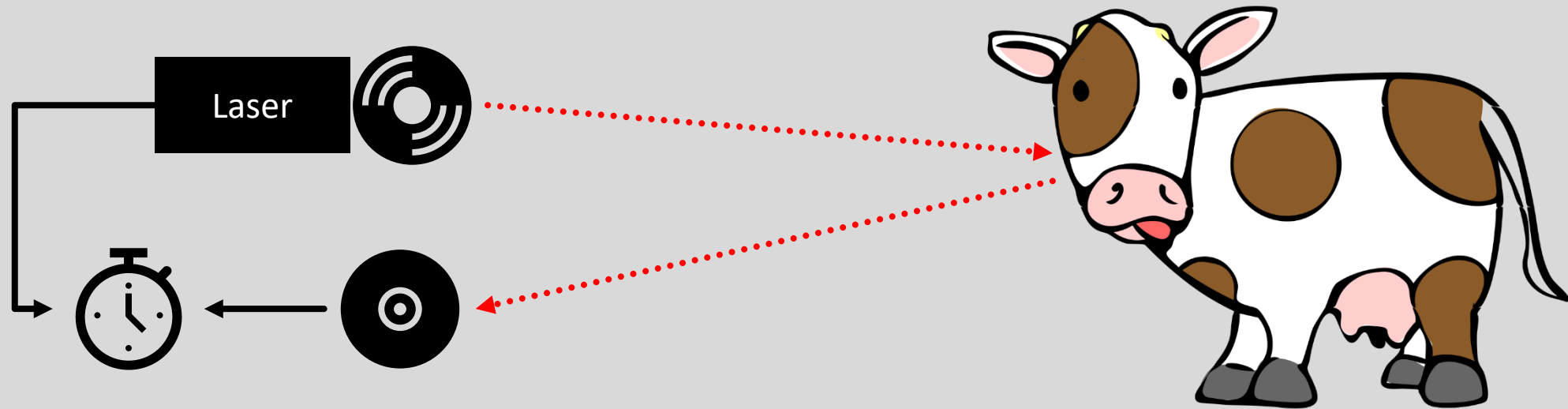
The screenshot displays the MATLAB/Simulink Test Manager interface. The main window shows the Test Manager tool with a test case named 'Plus2Auto_plausibilization'. The test results are displayed in a table with columns for NAME, STATUS, and a color-coded indicator. The test results show a 'Plausibilization Test' with a status of '1' (pass) and a 'BRAKE_SIG' property with a value of 'BRAKE_SIG'. A graph shows the test results for 'Signal A' (blue), 'Signal B' (red), and 'BRAKE_SIG' (green) over time. The Requirements Editor window is also visible, showing a list of requirements with columns for Index, ID, Summary, Verified, and Implemented. The requirements are organized into a tree structure, with the 'plus2auto' requirements expanded. The Properties window on the right shows the file path, revision, and creation/modification dates for the test case.

NAME	STATUS
Results: 2020-Apr-29 13:33:06	1
Results: 2020-Apr-29 13:58:56	1
Results: 2020-Apr-29 14:00:21	1
Results: 2020-Apr-29 14:13:33	1
Results: 2020-Apr-29 14:15:33	1
Results: 2020-Apr-29 14:17:01	1
Plausibilization Test	1
Baseline Criteria Result	1
BRAKE_SIG	1
External Inputs	1
Sim Output (Plus2Auto : norma)	1
BRAKE_SIG	1
Error:1(1)	1
BRAKE_SIG	1
Error:1(2)	1

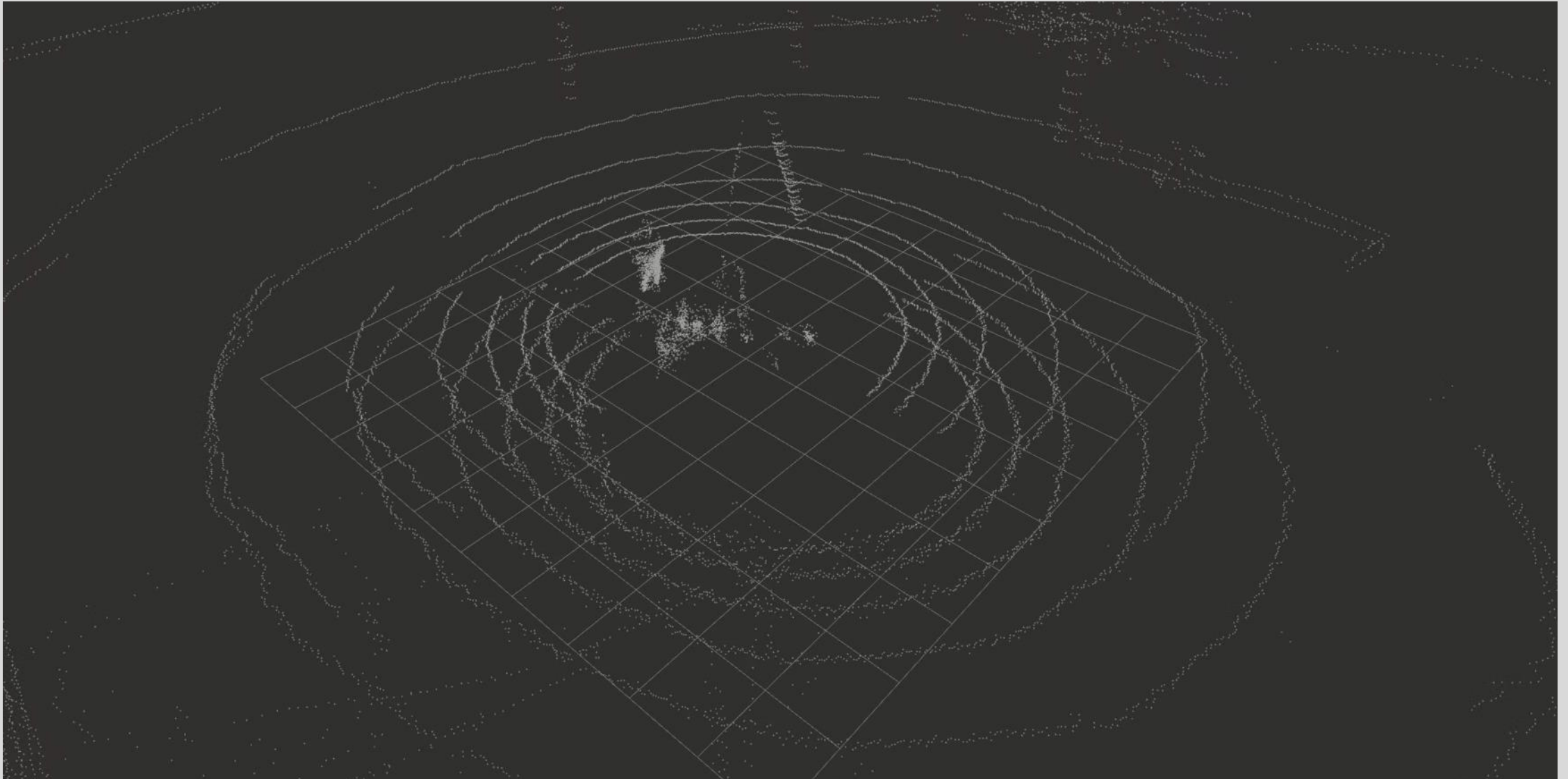
Index	ID	Summary	Verified	Implemented
1	#98	Simulation Tests		
2	#1	Input Handling		
2.1	#2	Input Validation		
2.2	#3	Input Scaling		
3	#4	Output Handling		
3.1	#5	Output Scaling		
4	#18	Measurement Handling		
4.1	#25	Measurement Validation		
4.2	#26	Measurement Scaling		
5	#24	Brake Control		
5.1	#87	Front brakes shall be controlled individually		
5.2	#88	Brakes shall implement a PID controller		
5.3	#100	Brake signal will be scaled by proportional gain		
5.4	#101	Brake control signal will be cut off under a threshold		
5.5	#115	Rear brakes shall be controlled individually, linear feed-forward		
6	#86	Brake Control Modeling		
7	#95	Brake control testing		
7.1	#96	Linear braking response		
7.2	#97	Brake pressure validation		
7.3	#99	Brake signal shall only be within the range 0-100% after validati...		
8	#106	Steering Control		
8.1	#107	Steering controller will be controlled via PID		
9	#128	Steering Torque Control		
10	#108	Steering Control Modeling		
10.1	#109	Only allow motor velocity to change position if enable and on ar...		
10.2	#110	Initial conditions are set at the midpoint of the steering block ra...		
10.3	#111	Steering conditions are integrated based on motor velocity		
10.4	#112	Velocity is converted from RPM to ticks/s		
11	#113	Light Control		
11.1	#114	When steering angle request or actual angle crosses threshold, ...		
11.2	#130	State indicator light is actuated by the state machine.		
11.3	#131	In Standby modes light is solid red		
11.4	#132	In error modes light is flashing red		
11.5	#133	In human modes light is yellow		
11.6	#134	In autonomous modes light is green		
12	#122	System States		
12.1	#126	System shall provision an initialization state		
12.2	#123	System shall provision a state for human driven modes		

- Unit testing linked with requirements closes the V

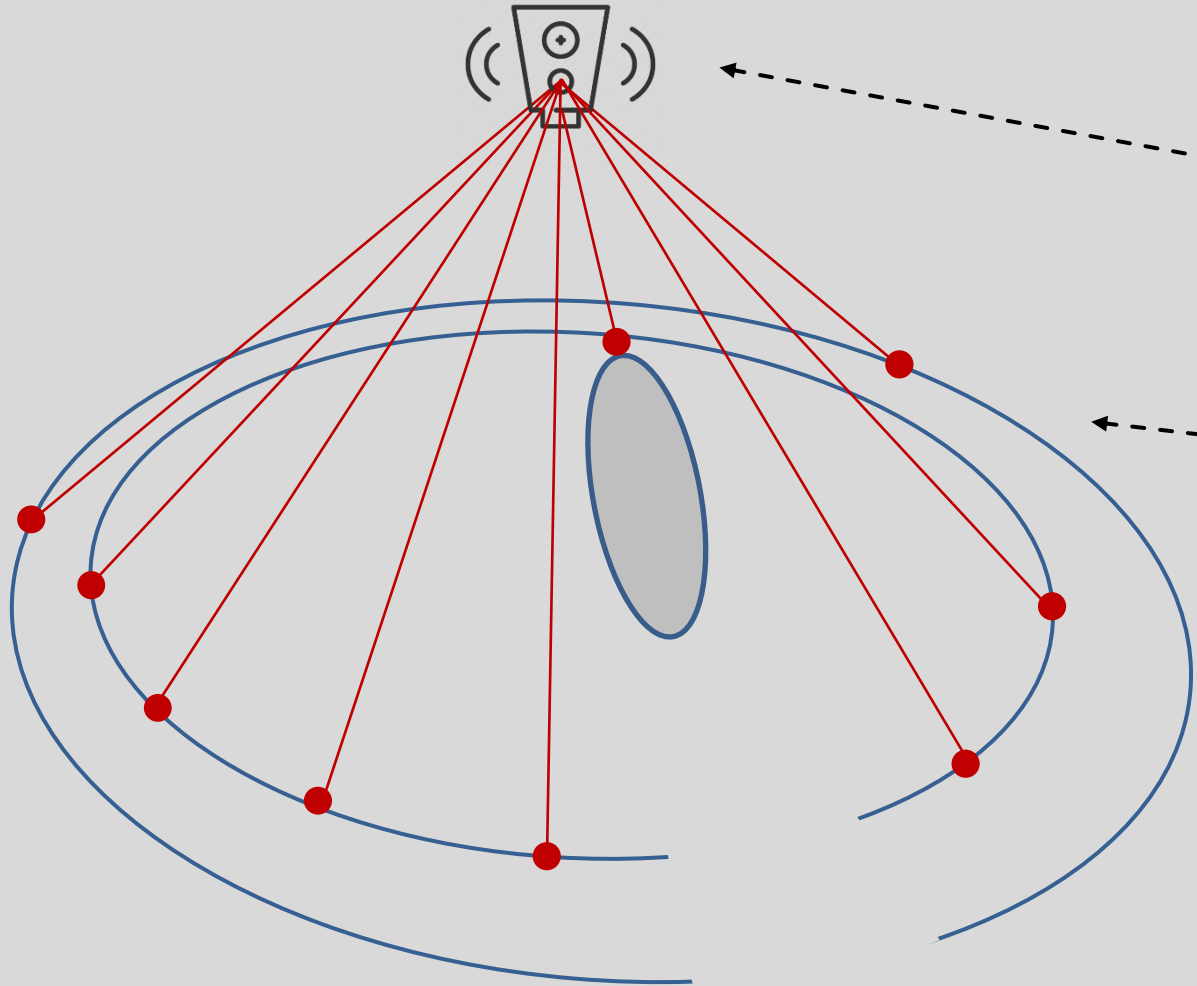
Safety Example: LiDAR (Light Detection and Ranging)



Complexity Management



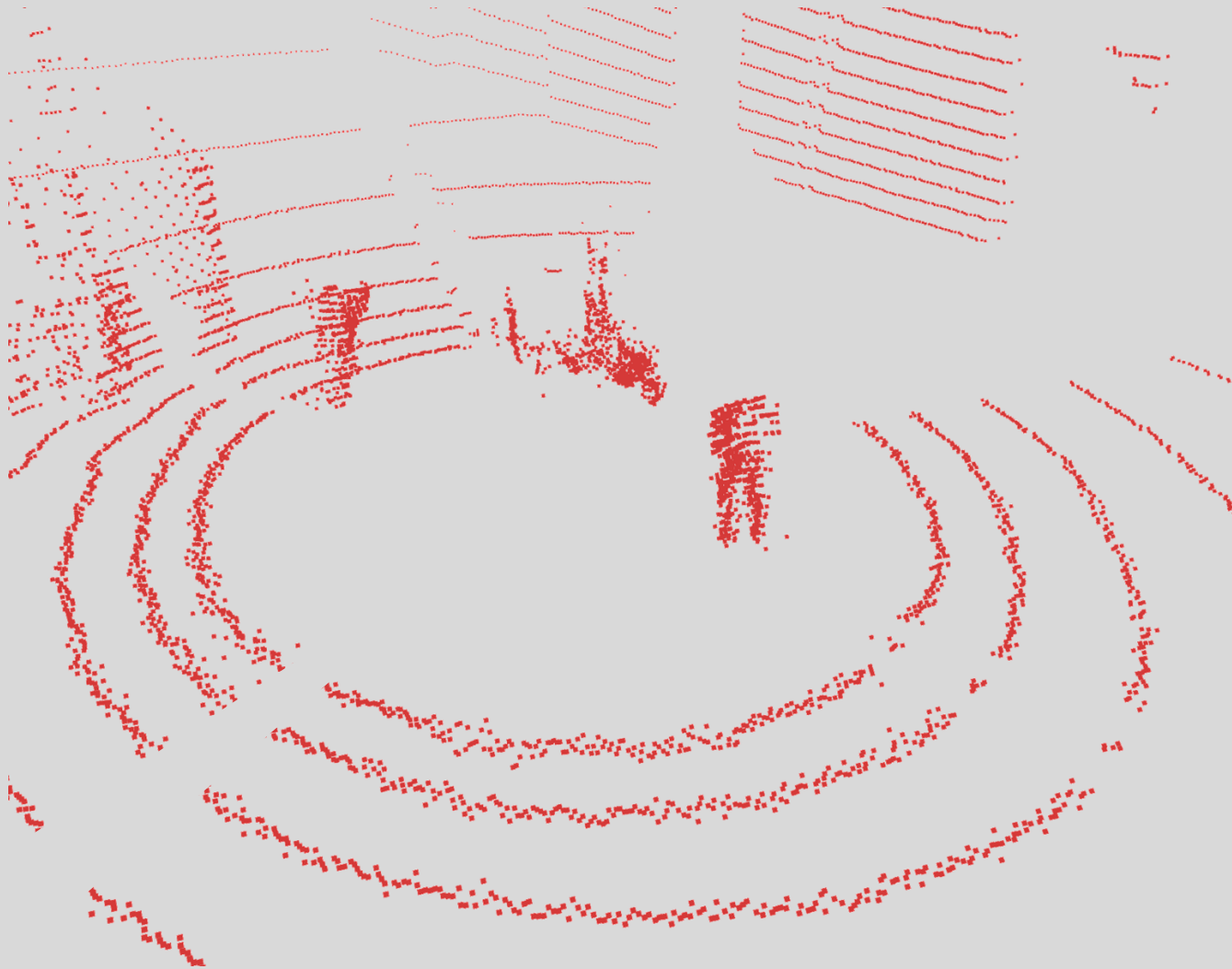
Internal laser reflections



Solution:

1. On even ground predictable rings are formed
2. Detectable obstacles modulate the ring form
3. Undetectable close objects create gaps in the rings

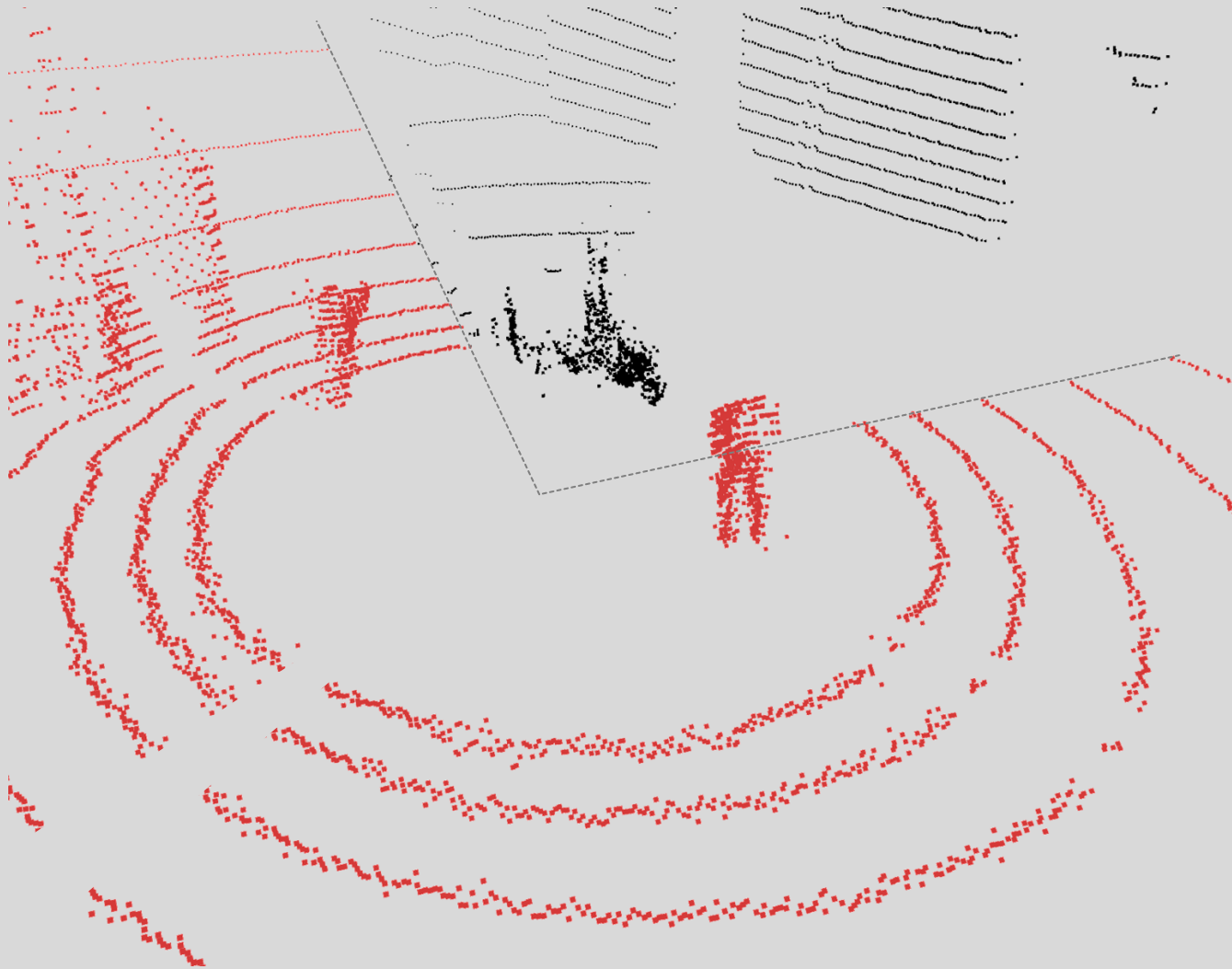
Near-field LiDAR object detection



Algorithm:

1. We receive a 16×1024 point cloud in 3 dimensions

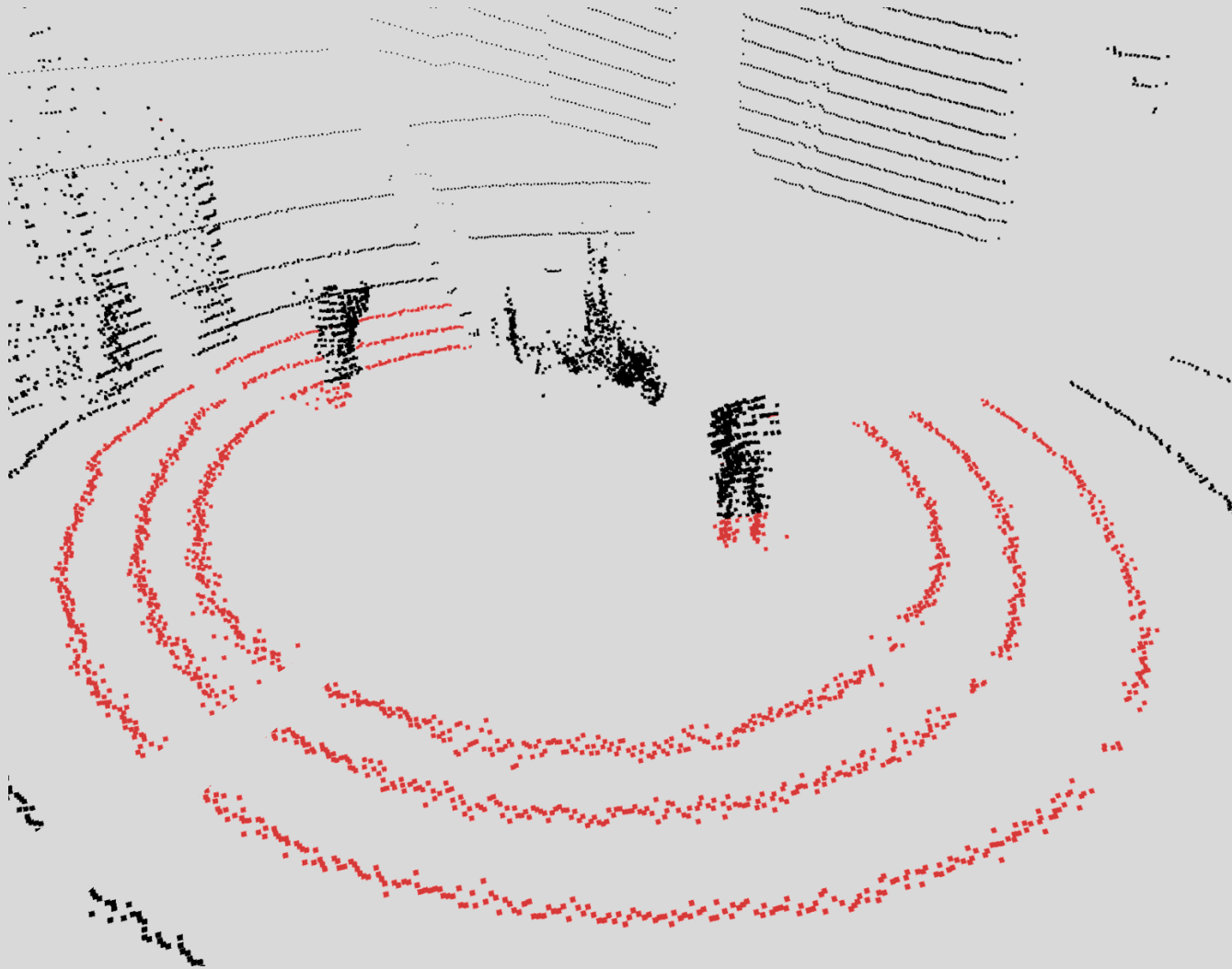
Objekterkennung mit LIDAR



Algorithm:

1. We receive a 16x1024 point cloud in 3 dimensions
2. We remove points which are echoed by the robot itself

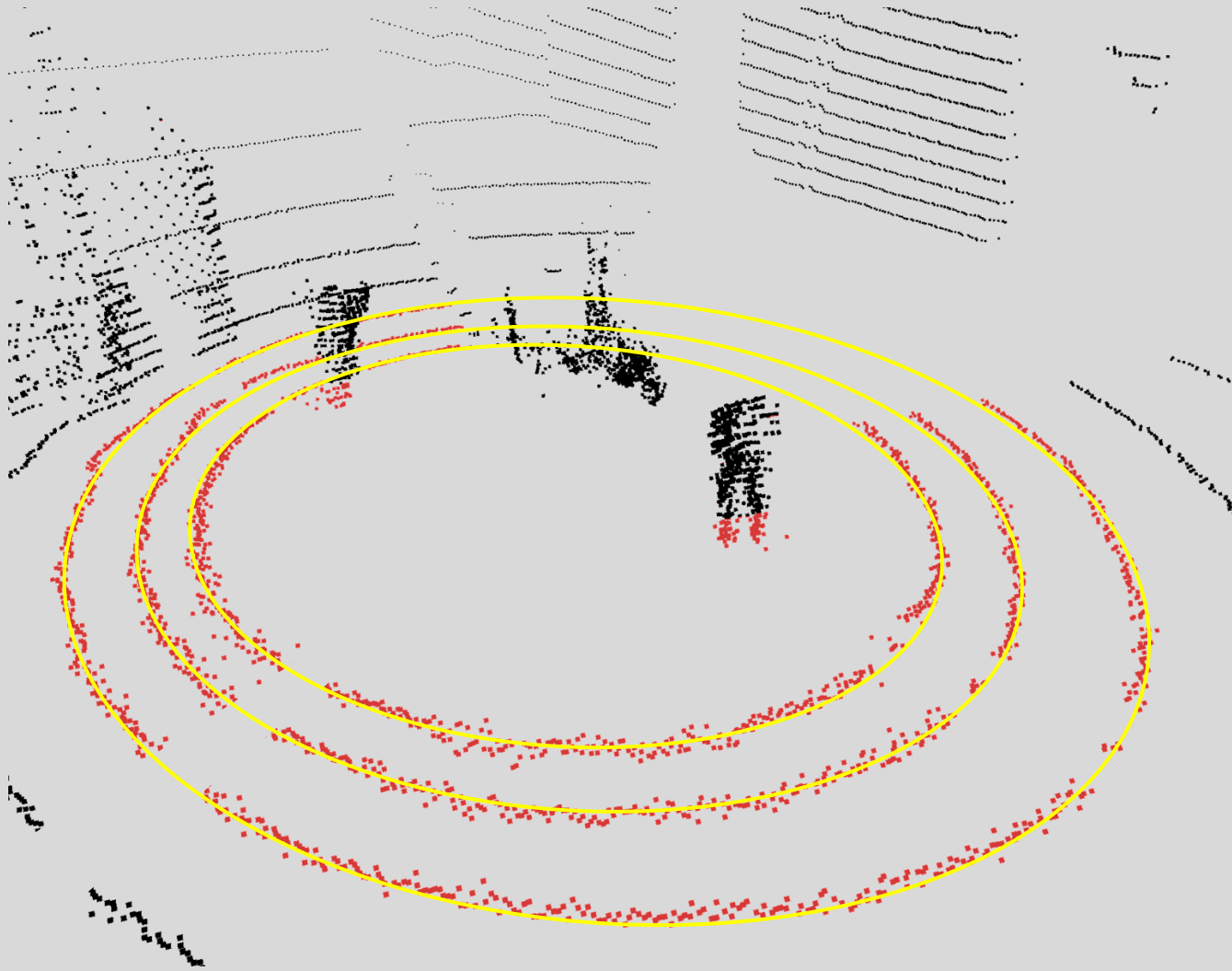
Objekterkennung mit LIDAR



Algorithm:

1. We receive a 16x1024 point cloud in 3 dimensions
2. We remove points which are echoed by the robot itself
3. We extract the lowest three rings

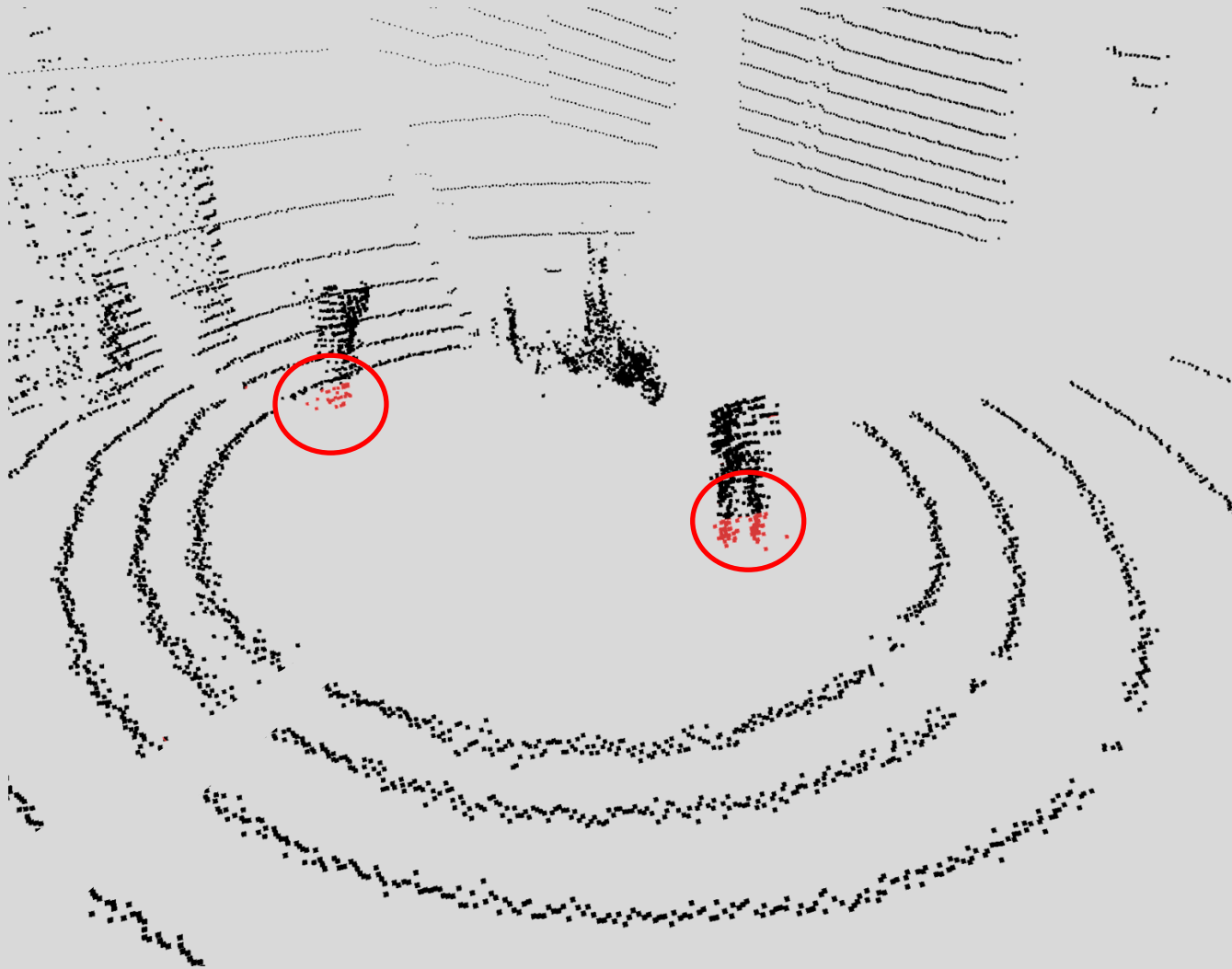
Objekterkennung mit LIDAR



Algorithm:

1. We receive a 16x1024 point cloud in 3 dimensions
2. We remove points which are echoed by the robot itself
3. We extract the lowest three rings
4. And search for ring shapes

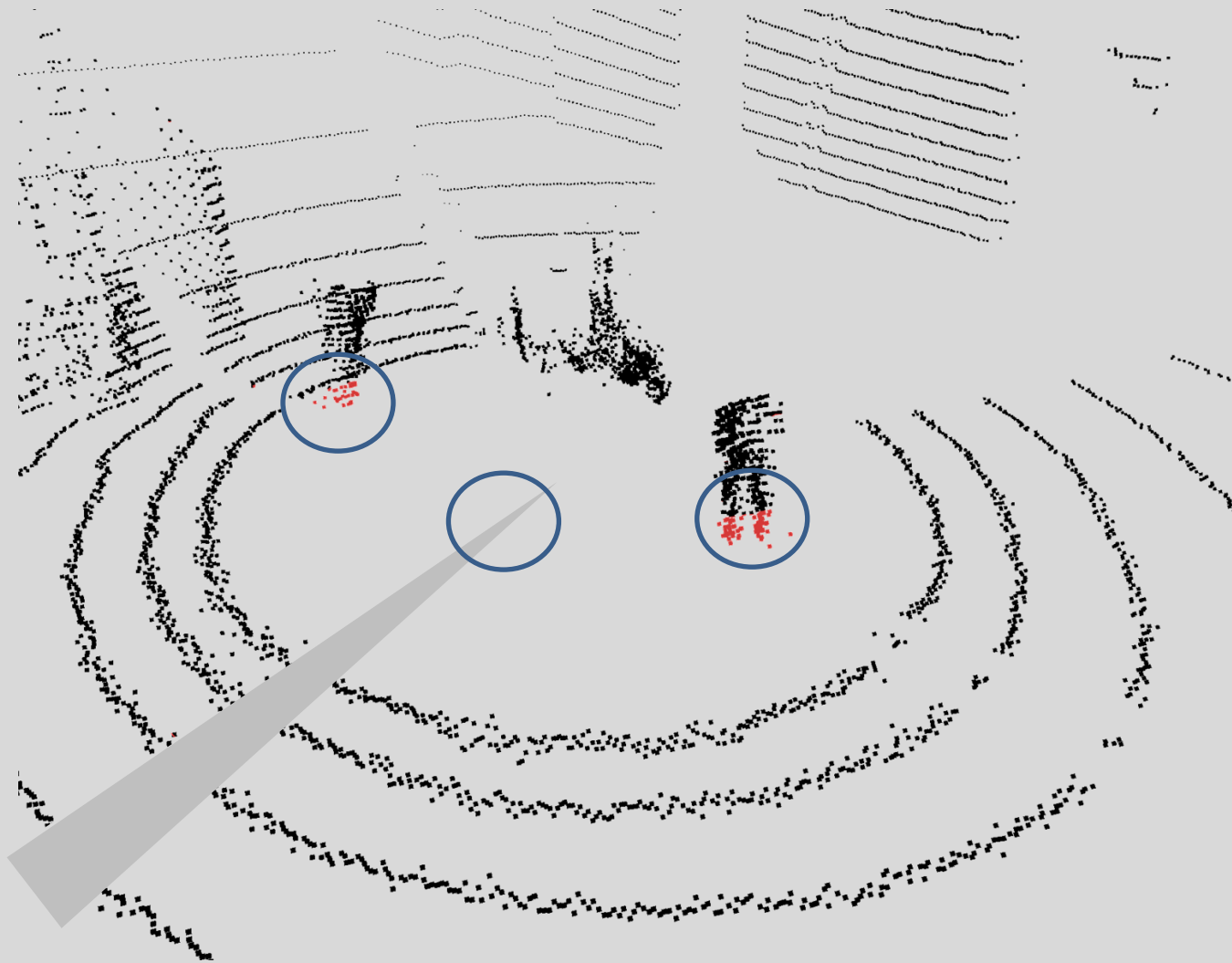
Objekterkennung mit LIDAR



Algorithm:

1. We receive a 16x1024 point cloud in 3 dimensions
2. We remove points which are echoed by the robot itself
3. We extract the lowest three rings
4. And search for ring shapes
5. Everything which is not part of a ring is classified as an obstacle

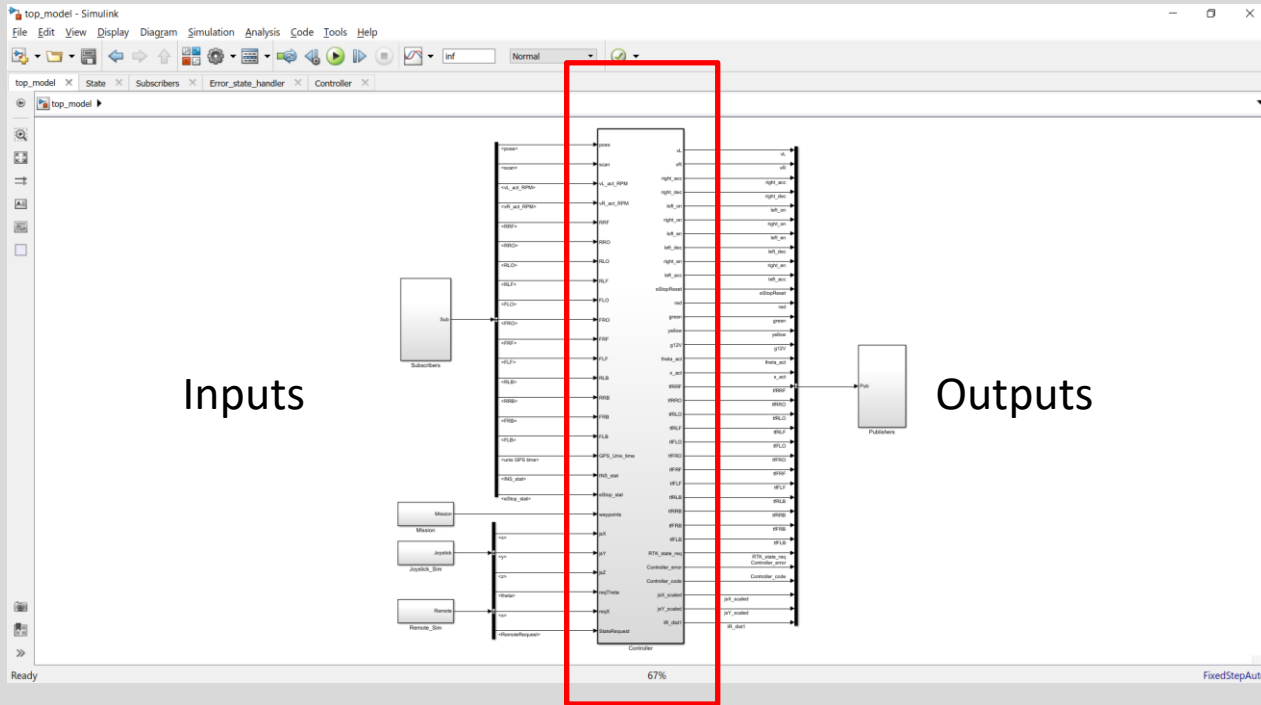
Objekterkennung mit LIDAR



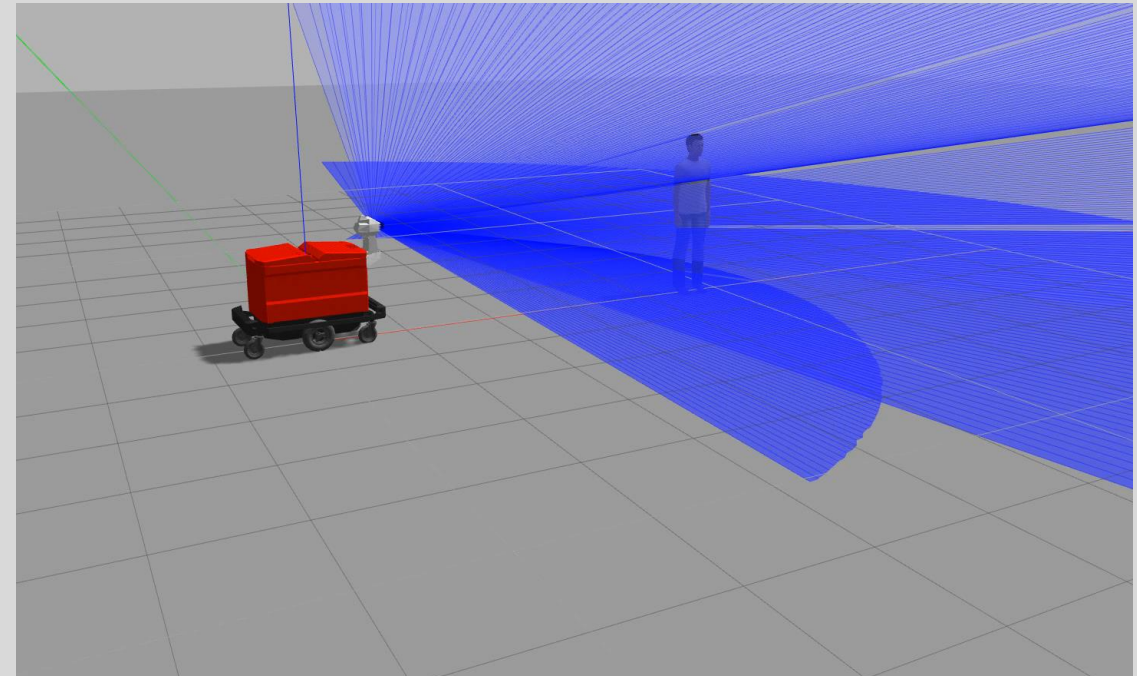
Algorithm:

1. We receive a 16x1024 point cloud in 3 dimensions
2. We remove points which are echoed by the robot itself
3. We extract the lowest three rings
4. And search for ring shapes
5. Everything which is not part of a ring is classified as an obstacle
6. Gaps in the rings are then also classified as obstacles

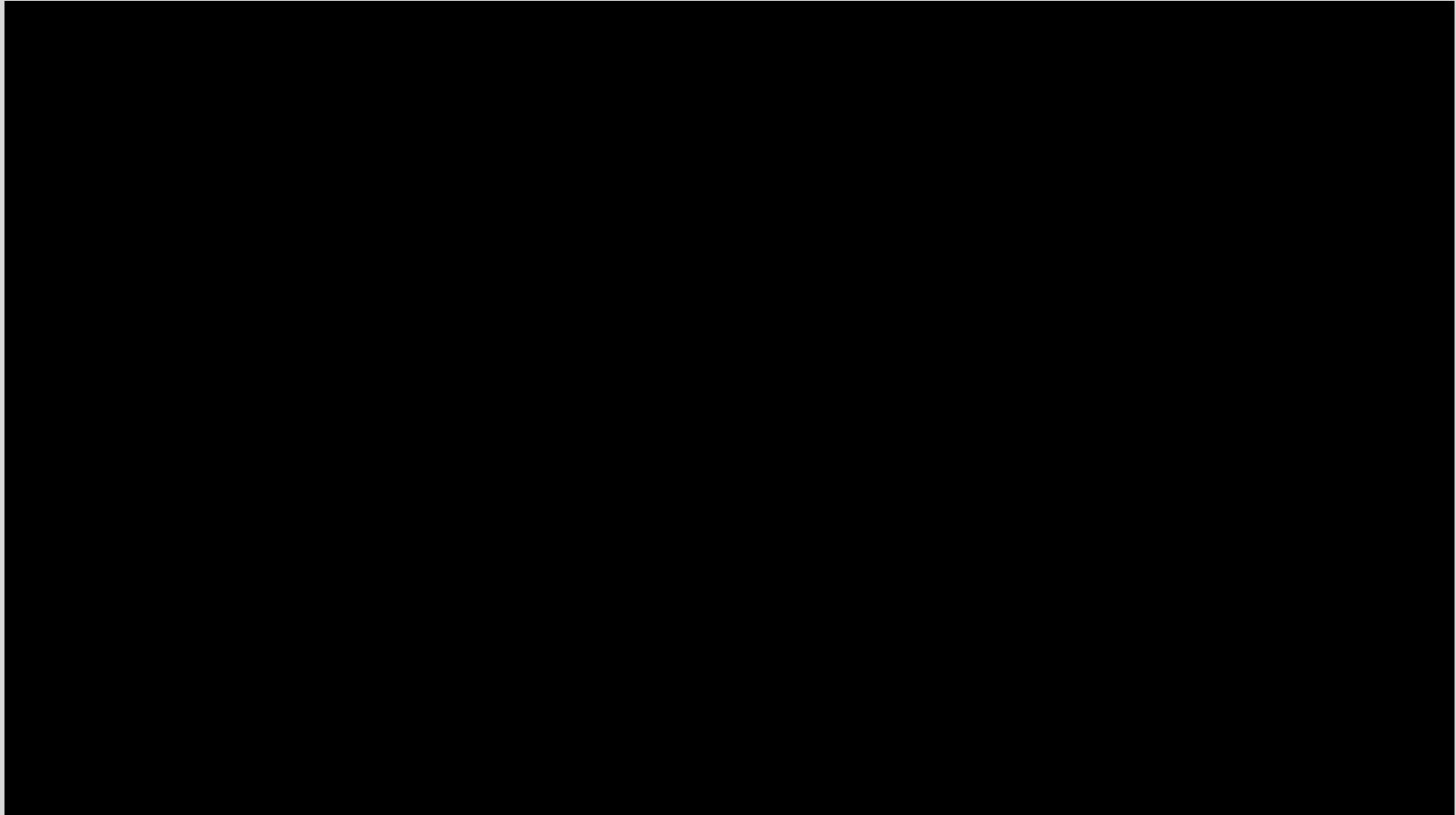
Simulation and Model-based Design



Controller Logic

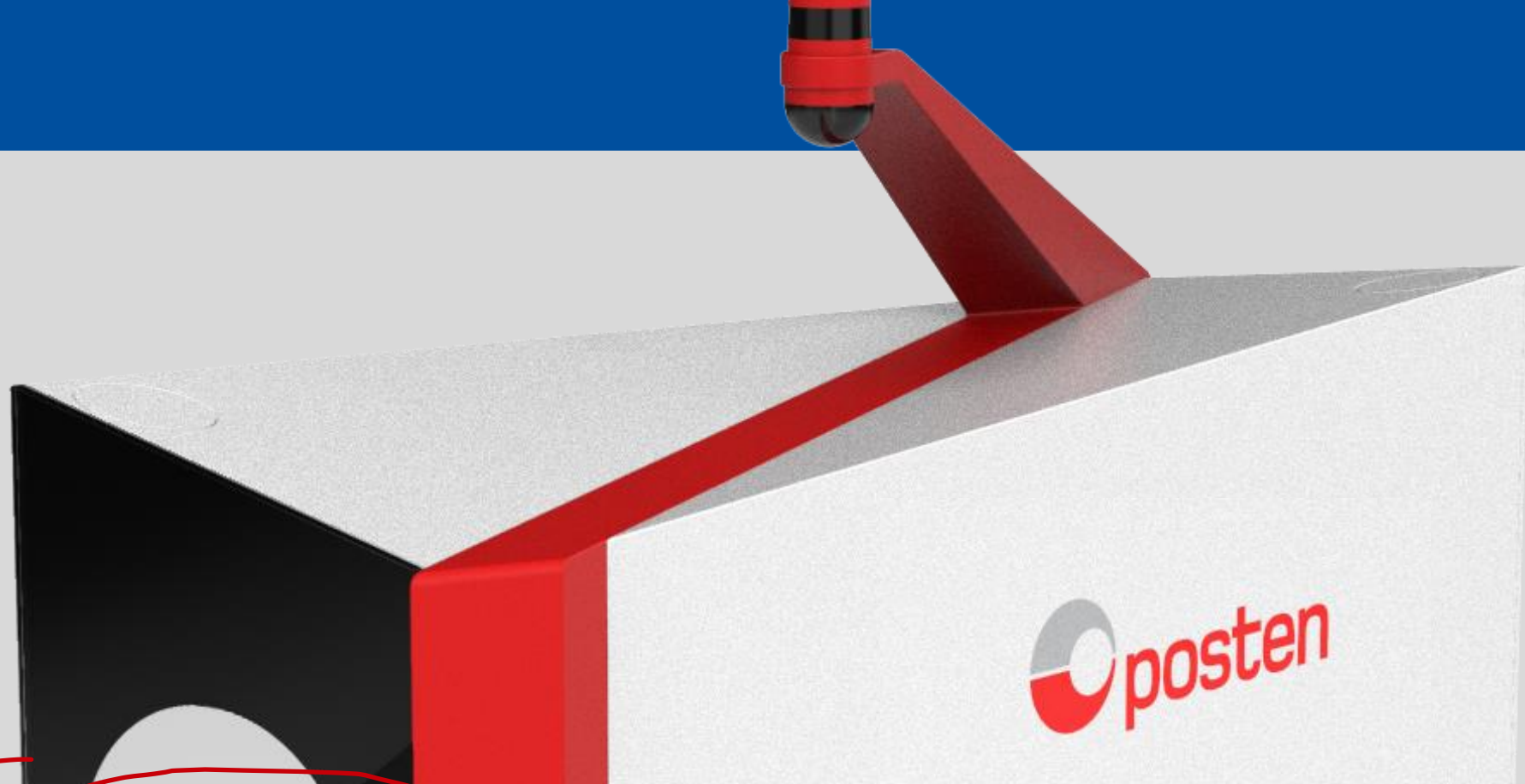


Vehicle in Action



Summary

- Kyburz was able to produce four functional prototypes in a span of 24 months using Model Based Design
- Some of the important lessons from this presentation:
 - Document and ensure adequate process is followed
 - Establish workflow which enables high safety integrity levels to be reached
 - Test at unit, module, and system level
- Success depends on effort multipliers (i.e. tools) as well as structured process
- Stay tuned for further deployments



Accelerating Deployment of Autonomous Delivery Robots using Model Based Design

Dr. Erik Wilhelm
KYBURZ Switzerland

11.04.2019

Q & A - Thank you for your attention

