

Projektgruppe FastSense

Meilenstein 2

Hardware Accelerated TSDF SLAM

28. Oktober 2020

Ziele und Anforderungen

- Ziele für MS2

- Funktionale Anforderungen

- Nicht-Funktionale Anforderungen

Implementierung

- Recap: Prototyping Demo

- Algorithmus

- FastSense Prototyp

- Kommunikation

Evaluation

- Strom

- Zeit

Fazit

- Verbesserungspotenzial

- Projektmanagement

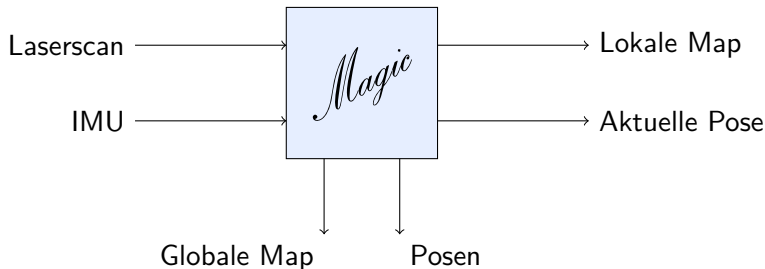
Ausblick / MS3

Ziele und Anforderungen

- Implementation von inkrementellem TSDF SLAM in „autarker“ Box
- Vorimplementation in Software
- Implementation einzelner Komponenten in Hardware
- Speicherung von Pose-Graph und TSDF-Karte zur Rekonstruktion des kompletten explorierten Bereichs
- Evaluation durch Zeit- und Strommessung

Funktionale Anforderungen

- Lokale TSDF-Map ausgeben
- Aktuelle 6D-Pose ausgeben
- Map auf Basis der IMU und Velodyne-Daten
- Trajektorie und TDSF-Map für jede Pose speichern
- Parameter zur Laufzeit anpassbar

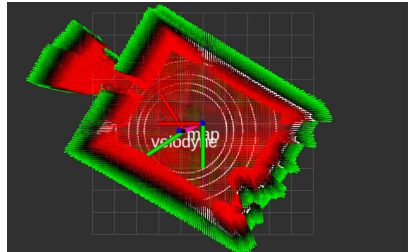
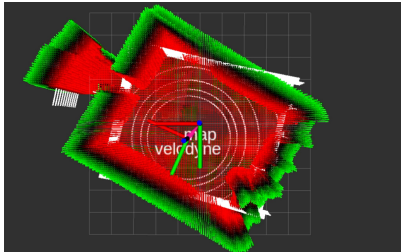


- HW-Plattform: Trenz-Board
- Entwicklungsplattform: Vitis
- Beschleunigung der Algorithmen durch FPGA
- Sensoren direkt am Board
- Unit-Tests
- Testbench
 - Integration, Strommessung, Zeitmessung, Visualisierung
- Logging

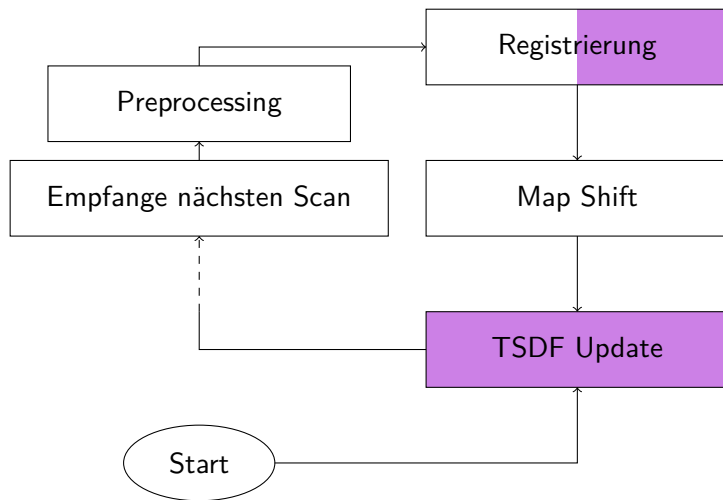
Implementierung

Recap: Prototyping Demo

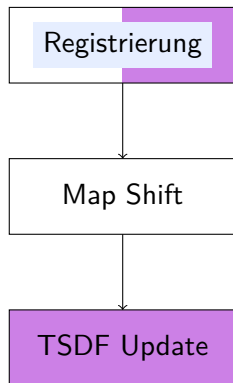
- Gute Parameterkombination herausgefunden
- Geplante Funktionalität war vorhanden und in RViz darstellbar
- Erkannte Probleme / Bottlenecks



Algorithmus



 = ausgelagert auf FPGA



T = Initial Transform;

Transform point cloud with T ;

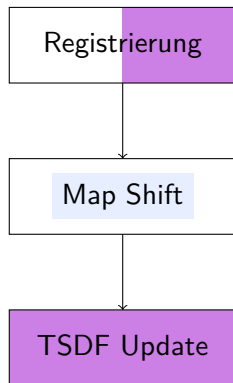
do

E = Minimization of the sum of
signed distances squared;

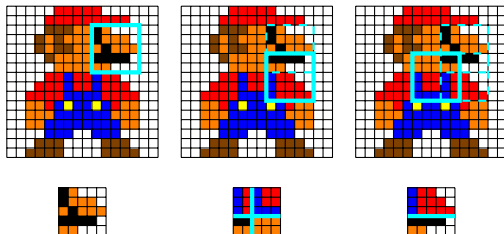
Transform point cloud with
 $T(E)$;

while *under max iterations and E
not converged*;

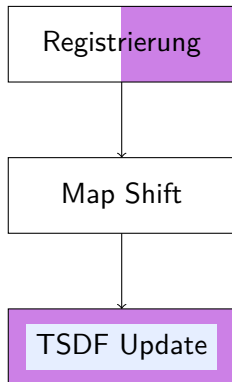
return T ;



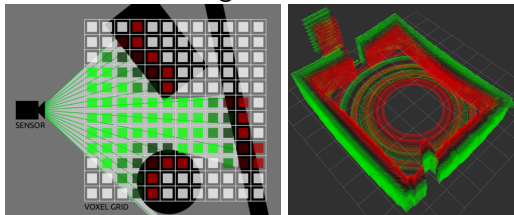
- LocalMap (aka RingBuffer)
 - Geteilt zwischen SW und HW



- Werte außerhalb in GlobalMap
 - Chunk-basiert
- Speicherung in HDF5



1. TSDF Generierung

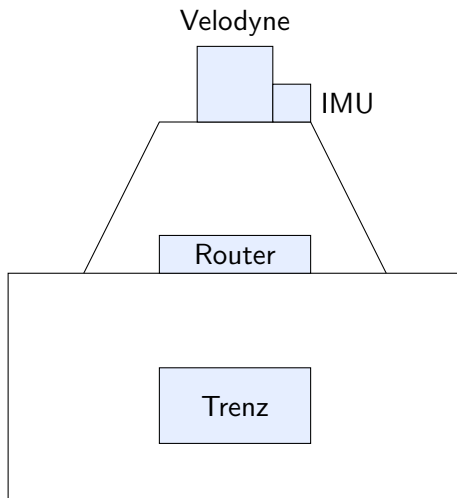


[1]

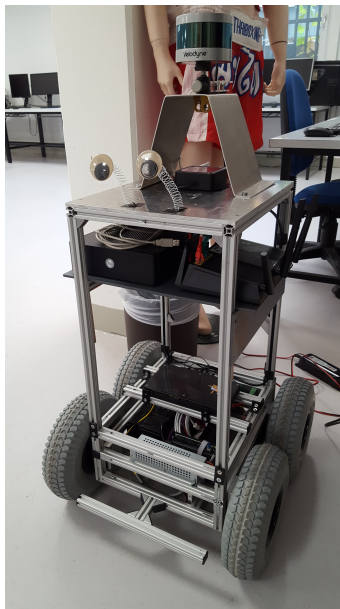
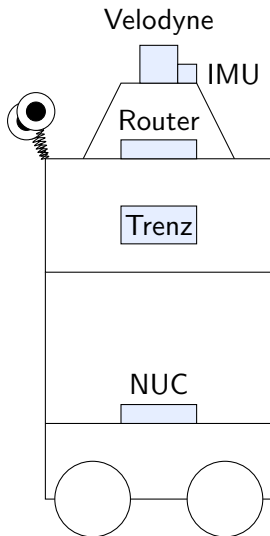
2. Update mit gewichtetem Mittelwert

$$M_V = \frac{M_V \cdot M_W + v \cdot w}{M_W + w}$$

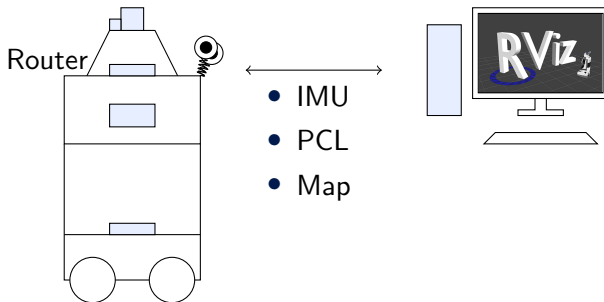
$$M_W = \min(M_W + w, W_{\max})$$



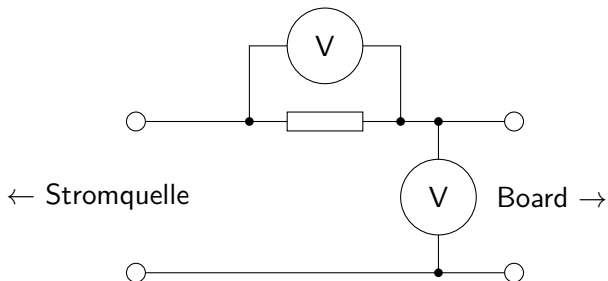
FastSense Prototyp



- Kommunikation via „Bridge“
- Datenaustausch über TCP, basierend auf ZeroMQ



Evaluation



TODO: Ergebnisse

Zeitmessung [ms]			
Abschnitt	Durchschnitt	Min	Max
Preprocessing	35	24	51
Registrierung	676	142	2894
Map Shift	105	0	1681
TSDf Update	337	319	346

Vergleich Vitis – Realität [ms]		
Abschnitt	Vitis	Gemessen
Registrierung	0.905	3.667
TSDf Update	124.483	552.924

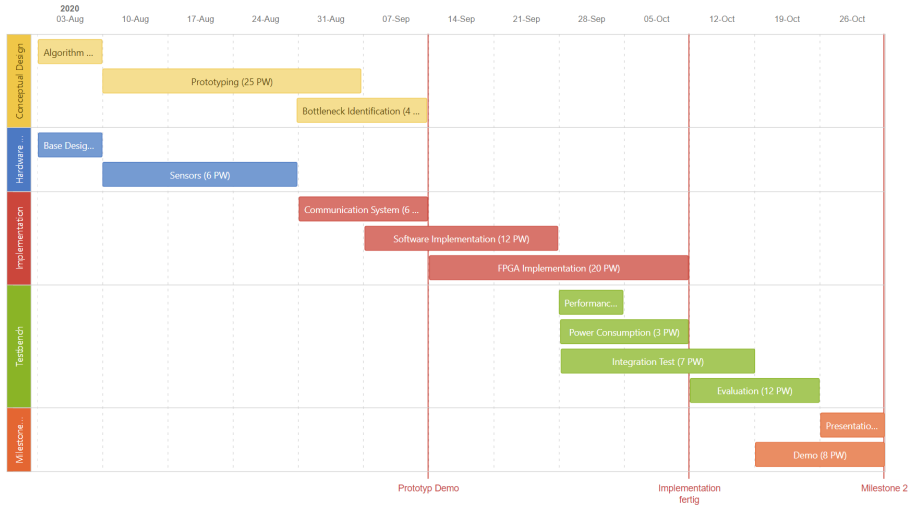
- Bei der Registrierung wird die meiste Zeit auf Speicher gewartet (95%)

Vergleich verschiedener Plattformen [ms]			
Programm	FastSense	Prototyp	Prototyp
System	Board	NUC	Stand-PC
Preprocessing	35	???	???
Registrierung	676	???	???
Map Shift	105	???	???
TSDF Update	337	???	???

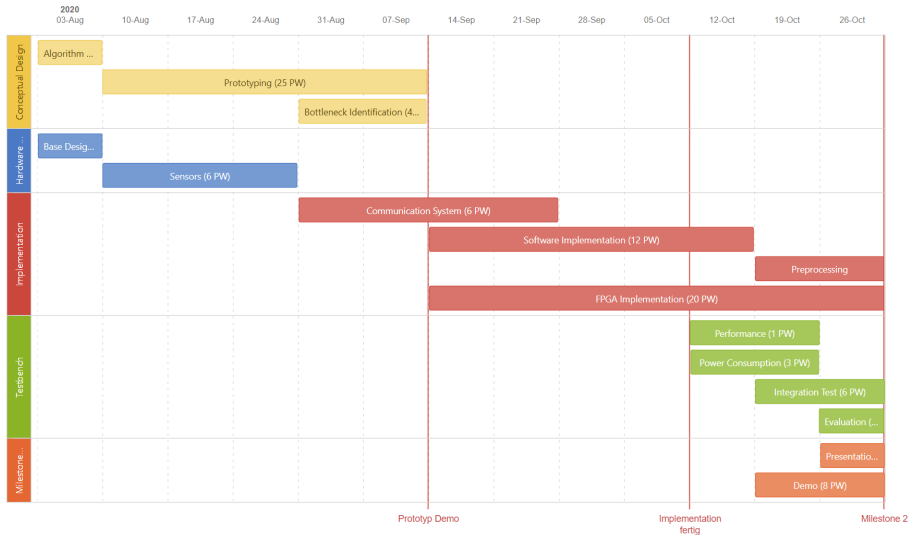
Fazit

- Registrierung
 - Drift entfernen (aktuell noch leichter Drift (1cm/s) in alle 3 Richtungen)
 - Komplette in Hardware (Overhead ist fast dreimal so hoch wie der eigentliche Aufruf)

Projektmanagement



Projektmanagement



- Aufbau einer SLAM-Box mittels CAD
 - Nutzung als Sensor
 - Einfache Portierung zwischen Drohne, Roboter, Rucksack etc.
 - Festes Interface, einfache Bedienung, Kapselung
- Verbesserung und Optimierung des Algorithmus
- Mesh-Generierung auf Basis der TSDF Werte
- Loop Closing
- ???

TODO: Mehr Ideen für MS3

- [1] D. R. Canelhas, T. Stoyanov, and A. J. Lilienthal. “SDF Tracker: A parallel algorithm for on-line pose estimation and scene reconstruction from depth images”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 3671–3676. DOI: 10.1109/IRoS.2013.6696880.