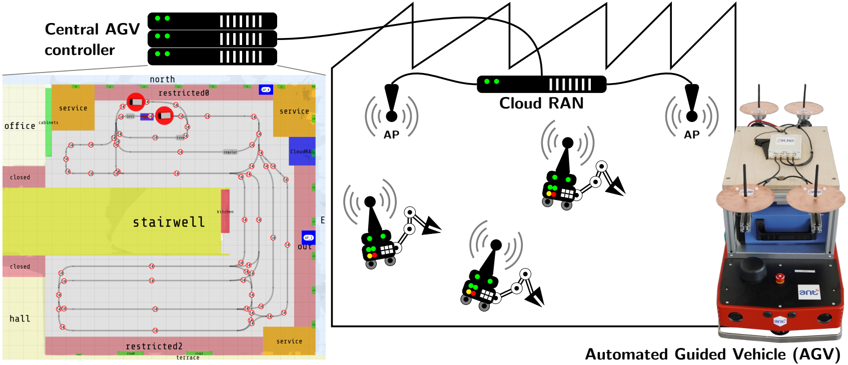
An ultra reliable low latency over-the-air communication system in GNU Radio for automated guided vehicles

Johannes Demel

Department of Communications Engineering
University of Bremen, Germany

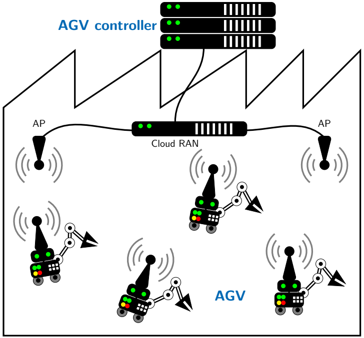
29th September 2022

Industrial Indoor Scenario - Factory Hall



Communication system requirements?

Communication system requirements



Ultra Reliable Low Latency Communication[1]

Ultra Reliable

Consecutive packet errors N_{sd}

$N_{\text{sd}} = 1$ OK

$N_{\text{sd}} = 2$ Critical

$N_{\text{sd}} = 3$ Fatal

Avoid burst errors!

Low Latency e2e $< 1\,\text{ms}$

real-time deadline

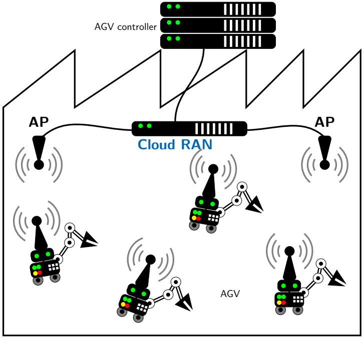
Periodic deterministic **Communication**

small packets $\sim 32\,\text{B}$ to $64\,\text{B}$

How to implement?

[1]Ultra Reliable Low Latency Communication (URLLC)

What do we want to implement?



Cloud RAN

Cloud

- Abstract compute resources
- Software paradigm
- Explore platform specifics
- Characterize DSP latency

RAN

- Distributed Access Points (APs)
- Joint Digital Signal Processing (DSP)
- Verify technologies

Can we implement this system in GNU Radio?

AGV communication system

Current system

- WiFi at 2.4 GHz/5 GHz (ISM)
- Listen before talk

Our system

- GNU Radio system at 3.8 GHz (*n78*)
- Dedicated campus network

Application

- Currently 100 ms periodicity
- **Requirement:** Lower periodicity
- **Problem:** Current system too unreliable

Goals

- **Optimized communication system**
- **Target:** 20 ms periodicity
- High reliability
- Low latency
- Short packets

How to improve reliability?

Diversity

Time ✗

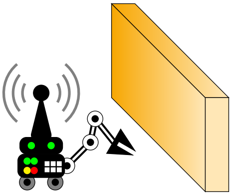
- Shadowing
- Blocking
- Coherence time $>$ latency requirement

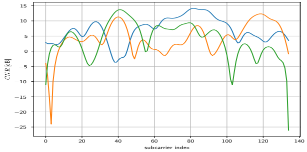
Frequency ✗

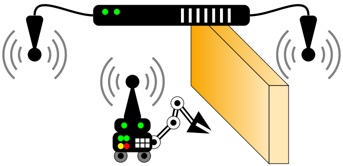
- Always Rayleigh fading
- Large coherence bandwidth

Spatial ✓

- Mitigate shadowing & blocking
- Independent fading paths



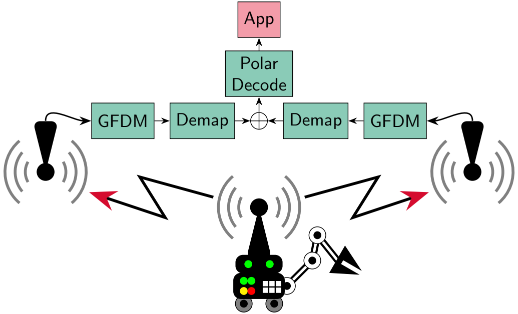




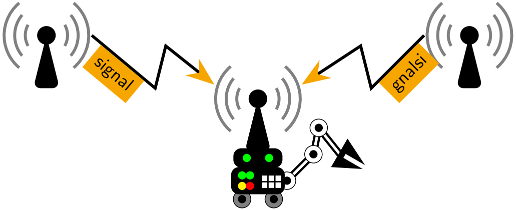
[0] Düngen et al. "Channel measurement campaigns for wireless industrial automation"

Spatial Diversity

Uplink: Joint decoding

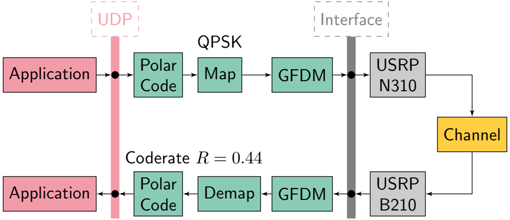


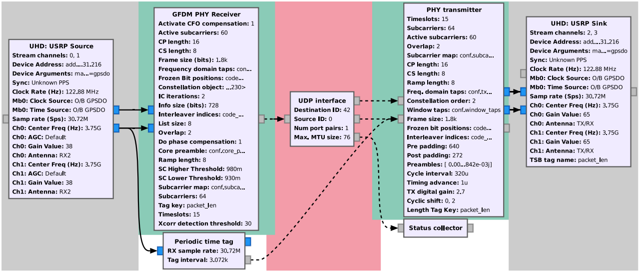
Downlink: Cyclic Delay Diversity



Approach: Distributed access points and joint processing

Demo implementation





Application

- Linux Ethernet interface
- → Foo-Over-UDP
- Socket PDU block (UDP)

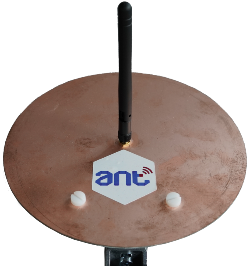
DSP

- Software-only
- Polar codes
- QPSK mapping
- GFDM multicarrier

Frontend

- N310 via 10 Gbit Ethernet
- B210 via USB

Base station with distributed access points



Antenna

- 3.3 GHz to 3.8 GHz (Pasternack PE51084)
- 2×2 MIMO
- Custom ground plane
- Antenna rack

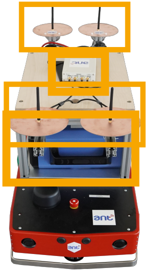
Distributed AP

- N310 connected via Fiber
- Octoclock sync with 10 MHz reference & 1PPS
- 30.72 MS s^{-1} sample rate

CloudRAN

- 32-core Ryzen 3970X
- 64 GB RAM

Mobile communication hardware



Peripherals

- same antenna with custom ground plane
- 2×2 MIMO
- Jauch JES1500WHA

Frontend

- B210
- Connected via USB
- same sample rate $30.72\,\mathrm{MS\,s^{-1}}$

Computing

- 12-core Ryzen 5900X
- 32 GB RAM
- Connection to on-board PC

GNU Radio benefits

Multi-threaded scheduler

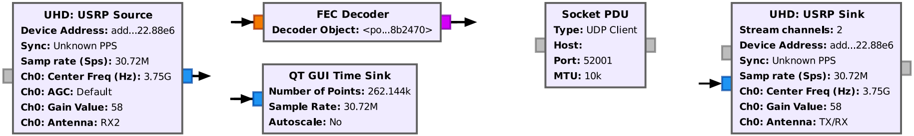
"It's like two drunkards trying to share a beer. ...
Sooner or later, they're going to get into a fight." ZeroMQ by Pieter Hintjens

Infrastructure

- Qt GUI
- Hardware blocks (USRP)
- Networking (UDP etc.)

Community

- Helpful people **Thanks!**
- Huge contributor base



Implemented Out-of-Tree modules

Polar Codes

- Good for short packets
- Very fast implementation
  `ant-uni-bremen/polar-codes`

Standardized symbol mapping

- LTE/NR/WiFi constellations
- Upstreamed block interleaver

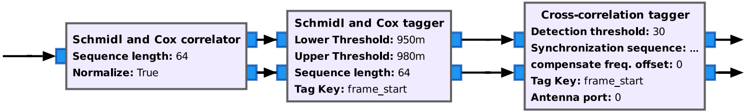
`ant-uni-bremen/gr-symbolmapping`

GFDM multicarrier

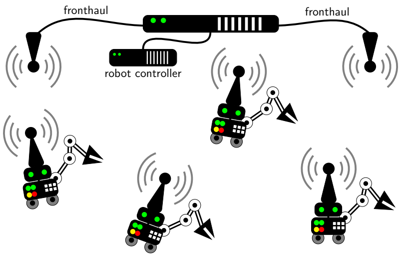
- Low latency implementation `jdemel/gr-gfdm`
- Fast synchronization `jdemel/XFDMSync`

Support modules

- Latency measurements
  `ant-uni-bremen/gr-latency`



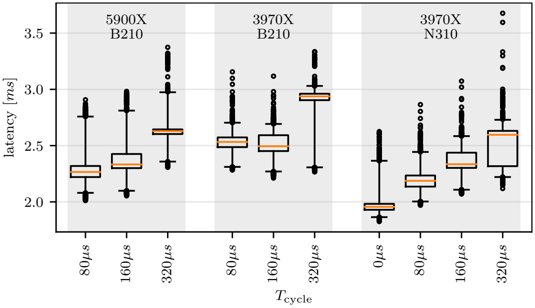
Demo parameters



- carrier frequency 3.75 GHz
  - ➔ campus network
- sample rate 30.72 $\mathrm{MS\,s^{-1}}$
  - ➔ B210 maximum for 2×2 MIMO
- bandwidth 29.28 MHz
  - ➔ beyond LTE
- 56 B UDP payload ➔ 99 B frame
  - ➔ small packets
- Over-the-air packet duration $\sim$37 µs
  - ➔ Spare resources for TDM

Latency measurements

Round trip time (RTT)



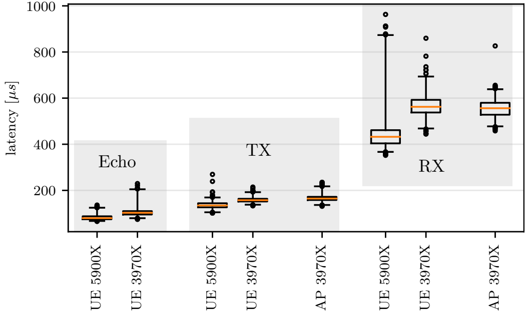
- Two CPU generations (3970X, 5900X)
- Two USRPs
  - B210 520 µs timing advance (USB)
  - N310 320 µs timing advance (Ethernet)
- Timed TX every T_{cycle}

Application requirement: 20 ms **periodicity** ✓

Latency measurements

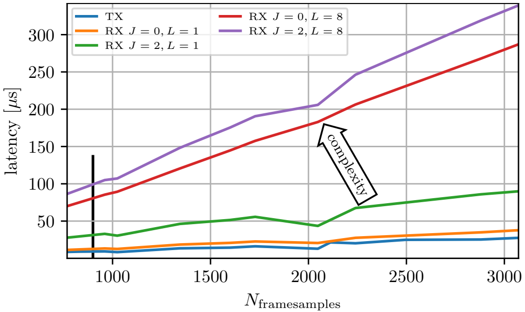
DSP

GNU Radio flowgraph measurements



Echo 82 μs to 107 μs
TX 135 μs to 166 μs
RX 439 μs to 565 μs

DSP function measurements

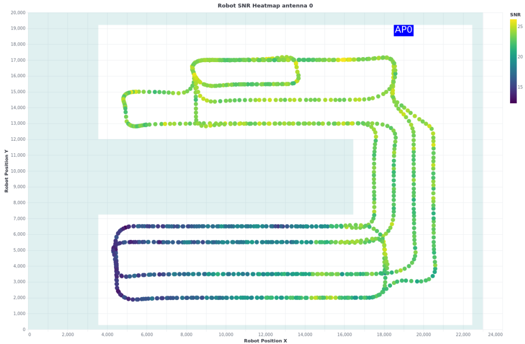


TX 9.3 μs
RX
- fastest 13 μs → $J = 0, L = 1$
- slowest 104 μs → $J = 2, L = 8$

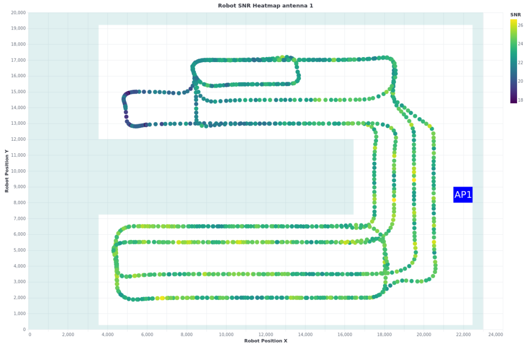
Conclusion: Huge multi-threading overhead

SNR measurements

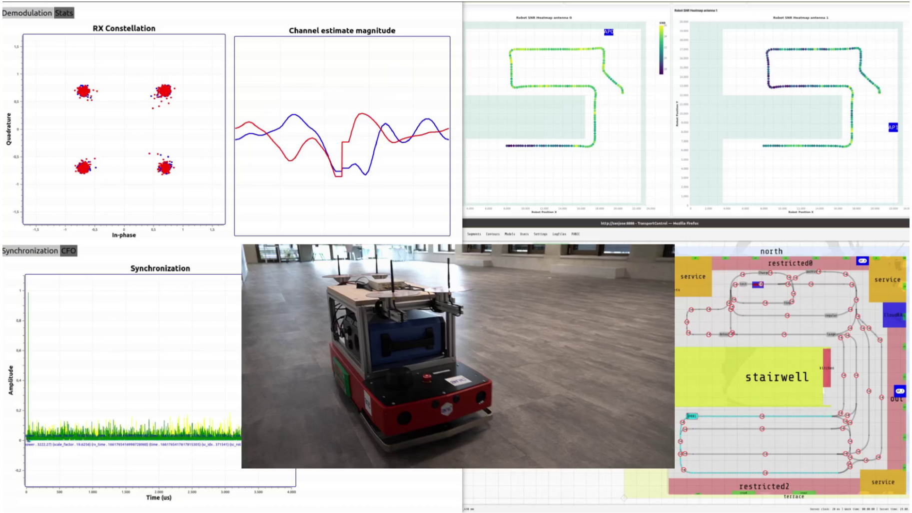
Robot SNR heatmap antenna 0



Robot SNR Heatmap antenna 1



Video demo


Demodulation Stats
RX Constellation
Quadrature
In-phase
Channel estimate magnitude
Synchronization CFO
Synchronization
Amplitude
Time (us)
north
restricted0
service
service
stairwell
out
restricted2
service
terrace

Conclusion

Recap

- AGV system with periodic communication
- URLLC requirements
- Industrial Radio environment
- Latency and SNR measurements

Results

- GNU Radio implementation available
- Application requirements fulfilled
- High reliability
- Low latency

Conclusion

GNU Radio can support highly reliable and low latency communication!

We are hiring!

We're hiring!

applications@ant.uni-bremen.de

More information at www.ant.uni-bremen.de

Resources

git repositories

- https://github.com/ant-uni-bremen/gr-symbolmapping
- https://github.com/ant-uni-bremen/gr-latency
- https://github.com/ant-uni-bremen/polar-codes
- https://github.com/jdemel/gr-polarwrap
- https://github.com/jdemel/gr-gfdm
- https://github.com/jdemel/XFDMSync