INSTITUTE FOR REAL-TIME COMPUTER SYSTEMS
TECHNISCHE UNIVERSITÄT MÜNCHEN
PROFESSOR SAMARJIT CHAKRABORTY

Radar Reprojection Mapping Improves Obstacle Avoidance in Mobile Robots with an Unsteered Radar Sensor

Laurenz Altenmüller

Master's Thesis

Radar Reprojection Mapping Improves Obstacle Avoidance in Mobile Robots with an Unsteered Radar Sensor

Master's Thesis

Supervised by the Institute for Real-Time Computer Systems
Technische Universität München
Prof. Dr. sc. Samarjit Chakraborty

Executed at BSH Home Appliances Corporation
Bosch Research and Technology Center
Palo Alto, California, USA

Advisor: Michael Balszun, M.Sc.

Author: Laurenz Altenmüller
Pommernstr. 9
80809 München

Submitted in September 2017

Abstract

Mobile indoor robots mostly rely on lidar and vision sensors to remotely detect obstacles in their path. These sensors have trouble detecting some common real-world obstacles like transparent surfaces and chair legs. Recent affordable near-range miniature radar sensors enable new solutions. This thesis explores a simple solution for a radar sensor being moving through a static environment without the need for beamforming or a mechanically scanning radar. The idea is put to test in experiments with an Omniradar FMCW radar mounted on a Kobuki robot platform. A comparison against lidar and RGBD sensors shows how promising the setup is for navigation in unstructured environments.

Acknowledgements

Vielen Dank . . .

München, im Monat Jahr

Contents

List of Symbols

Symbol	Meaning	Dimension
A_r	Effective antenna aperture	
c	Speed of light	

1 Theoretical Background

A variety of sensors enable robots to sense their environment. Most robots monitor information like temperature or track odometry from intertial measurement units (IMU) and wheel encoders. This section focuses on how a robot can avoid or detect collisions.

1.1 Traditional Obstacle Sensors

The fallback option of obstacle sensing is almost always a bumper sensor. If the robot runs into an obstacle in its path, a bumper switch will be pressed and the robot's navigation system knows that it can not traverse further in this particular direction. The concept can be extended to measure sudden acceleration that indicates a collision or even monitor motor current to find that the robot is stuck against an obstacle, but the idea is the same: With this kind of sensor, the robot can only detect that it already had a collision. The detection can be shifted to slightly earlier point in time with systems like whisker-like antennae or capacitive sensors[**Muhlbacher-Karrer2015**]. A special case is negative obstacle (e.g. cliff) detection. Here, usually Infrared (IR) range sensors are employed on the underside of the robot.

Intelligent navigation and path planning however can only be achieved with ranging sensors. Classic range sensors are IR, ultrasound (US), or laser based. They measure the distance to the closest target in one direction. Because obstacles can appear in any direction, these sensors need to be scanned. Scanning means that the bearing of active sensing direction is changed over time to achieve range scans that are multiplexed over a field of view (FOV). Usually this is done mechanically, with a sensor turret spinning on a servo motor. The best known example of this is the classic lidar sensor, which spins a laser range finder around.

Vision sensors are scannerless. Next to regular cameras as monocular vision sensors there are stereo-camera setups to record depth information. Structured light sensors such as the first-generation Microsoft Kinect sensor record depth disparity from triangulation, via correlation between a known and perceived projected light pattern. Time-of-flight (TOF) cameras like the second-generation Microsoft Kinect illuminate the scene with amplitude-modulated near-IR light and calculate depth from the phase shift between the transmitted and received signal[**Sarbolandi2015**].

There is also a scannerless version of lidar. Full-field lidar, also known as flash lidar [**Payne2008**] uses TOF measurements of omnidirectional light pulses to capture a 2D

scene.

To map out the environment and localize itself, the robot will usually employ a version of a simultaneous localization and mapping (slam) algorithm[**Cadena2016**].

1.2 Radar

Radar sensing is based on the transmission and reflection of electromagnetic signals. A transmitting antenna radiates EM energy, some of which is scattered off reflective objects, called targets, and intercepted by a receiving antenna. This signal is amplified and checked for time delay, frequency shift, phase shift and amplitude attenuation with respect to the transmitted signal. This allows to capture certain target properties like range, radial velocity, size, shape and, among others, even surface smoothness and orientation. [**Skolnik2008**]

The return signal's echo power is described for the interference-free case in vacuum by the radar equation,

$$P_r = \frac{P_t G_t}{4\pi R^2} \cdot \frac{\sigma}{4\pi R^2} \cdot A_r$$

where P_r received target echo power P_t transmission power G_t transmit antenna gain R range of target (distance) σ radar cross section (RCS) of target A_r effective area of receiving antenna. Of the three factors in the equation, the first factor represents the power density at the radar-illuminated target's distance. The second factor represents how much of the radar energy is scattered back by the target. The third factor finally denotes how much of the echo power is collected by the receiving antenna. [**Skolnik2008**] Conventional radar only becomes useful with directive antennas. The antenna gain G_t is defined as the ratio of increased power in a particular direction compared with that from an isotropic antenna [**Adams2012**]. Antenna theory shows [**Balanis2015**] the relation of receive antenna gain G_r with radiation wavelength λ:

$$G_r = \frac{4\pi A_r}{\lambda^2}$$

With constant antenna loss factor $L > 1$, substituting equation #REF into equation #REF yields the classical radar equation

$$P_r = \frac{P_t G_t G_r \lambda^2 \sigma}{(4\pi)^3 R^4 L} \propto \frac{\sigma}{R^4}$$

In practice, the actual received power is lower than predicted by this equation due to many factors, including interference and atmospheric conditions. σ is also not constant but varies with viewing angle and material properties of the target. [**Adams2012**]

1.2.1 Doppler effect

Many radar systems measure radial velocity with the Doppler frequency shift. Austrian physicist Christian Doppler described the kinematic effect in 1842. It describes the change of wavelength caused by the motion. A common example is the change of pitch that can be heard when a race car or ambulance passes the observer. The Doppler frequency shift f_D is

$$f_D = 2\frac{v_r}{\lambda} = 2\frac{v\cos(\theta)}{\lambda}$$

where v_r is the radial velocity component of the target, which travels at a speed v at angle θ between the target's direction and the radar beam with wavelength λ [**Skolnik2008**]. The factor 2 is caused by the Doppler shift being applied twice; once for the incident wave, and once for the reflected wave. In effect this means that a radar sending out an EM wave with a frequency of exactly $60GHz$ towards a target moving at a relative speed of $v_r = 1\frac{cm}{s}$ towards the radar will receive back an echo with a frequency of $60000000004Hz$ because of the frequency shift of $f_D = 4Hz$.

1.2.2 Types of Radar

Continuous Wave (CW) radar

CW radar was the first radar system. It uses a continuous transmission at a fixed frequency. Thanks to antenna directionality it can be used to find a target's azimuth in radio direction finding. A target's velocity information can be extracted from the frequency shift due to the Doppler effect.

Pulse radar

Pulse radars send series of short bursts. The time delay τ between transmission and reception of a pulse, called time of flight (TOF), gives a target's range R, so that

$$R = \frac{c\tau}{2}$$

The range resolution ΔR is given by

$$\Delta R = \frac{c\tau_m}{2}$$

with τ_m the pulse high time. To achieve high range resolution pulses must be very short, which requires very high peak transmission power to still produce a detectable echo signal. Pulse compression radars send a longer pulse with an internal modulation, which combines the higher transmission energy of longer pulses with the resolution of short pulses. Velocity is again known from frequency shift.

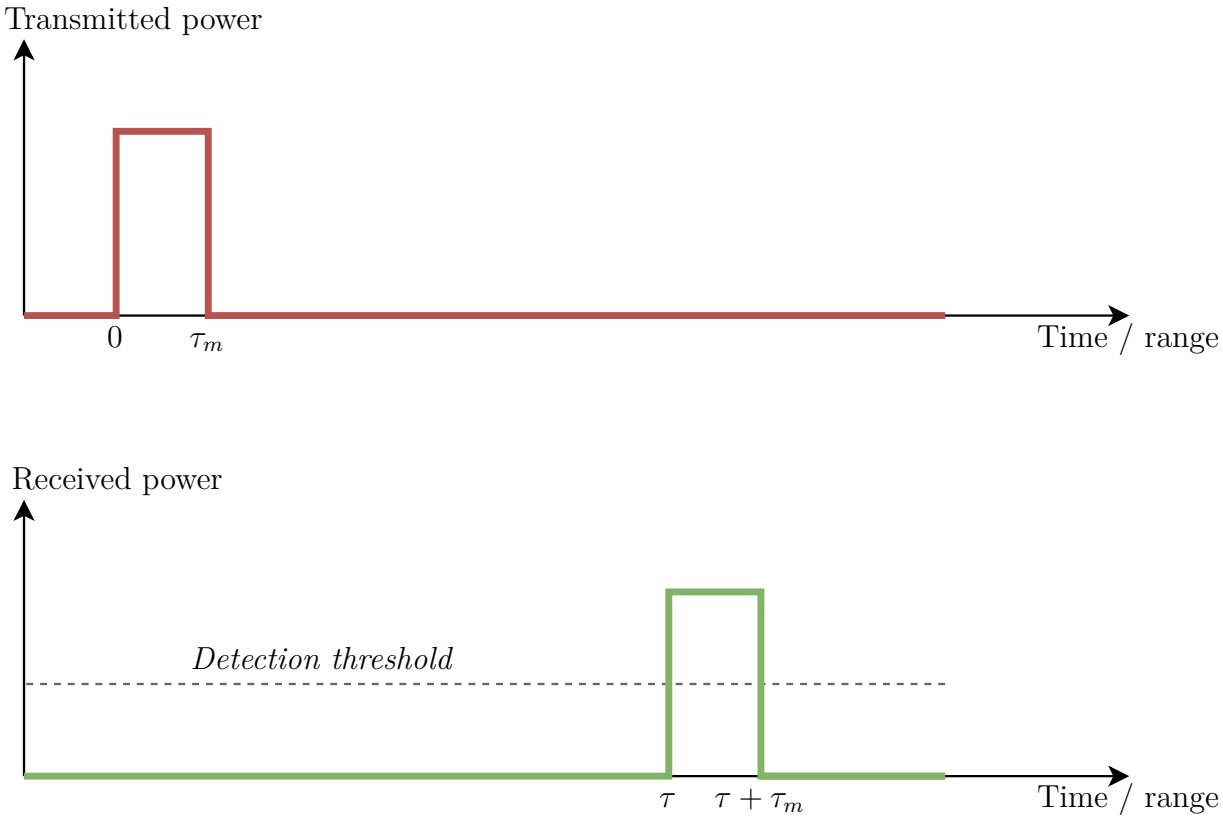


Figure 1.1: Pulse radars measure the time between stransmission and reception of a short EM burst. Adapted from [**Adams2012**] p. 52

1.2.3 Frequency-modulated continuous wave (FMCW) radar

FMCW radars use a frequency modulation to measure range and speed at the same time. The transmitted modulation is compared to the modulation in the received signal to detect signal delay and frequency shift. Applications in robotics use this kind of radar the most, for reasons of lower transmission power and high-range resolution [**Adams2012**].

An FMCW radar's modulation is called a frequency sweep or chirp and is usually triangular with a linearily increasing and decreasing frequency. Most sensor use a voltage controlled oscillator (VCO) to generate the modulation waveform. VCOs do not have a linear transfer function, so in order to obtain a linear sweep, the input to the VCO must be pre-distorted with the inverse of the VCO's nonlinear transfer function. Instead of a VCO, direct digital synthesizers together with phase-locked loops (PLL) can be used. They generate better (more linear) sweeps at the price of increased design complexity and cost [**Ernst2016**].

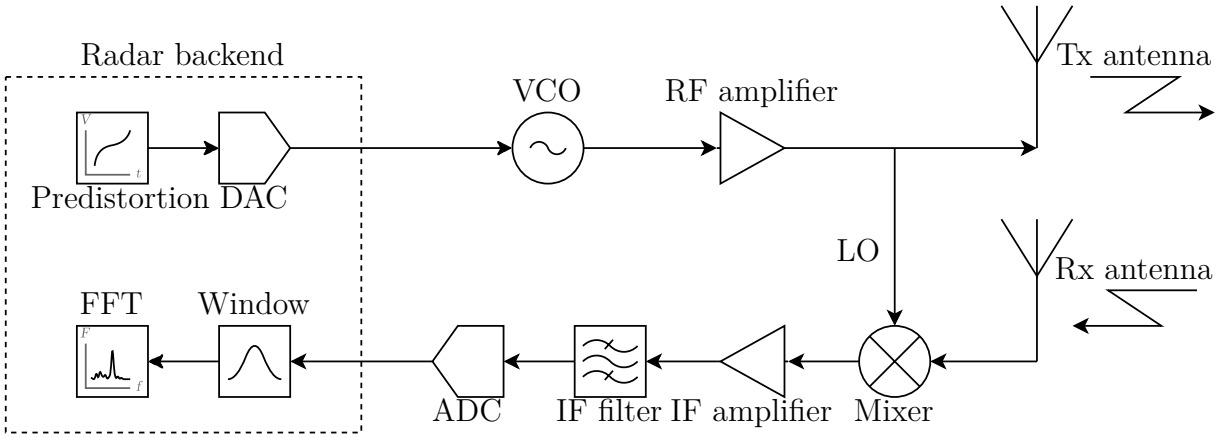


Figure 1.2: Simplified FMCW architecture. Adapted from [**VanZeijl2014**]

After the VCO's signal is amplified and transmitted, it reflects at visible targets and is received as echo in the same frequency band.

In the top subplot, figure **??** shows the transmitted frequency sweep from f_0 to $f_0 + \Delta f$ over a sweep length of T_d of a triangular modulation. The middle plot also shows the received signal as caused by a single stationary ideal reflector. Time of flight causes a delay τ in the received signal. To understand where the beat signal comes from, we focus on the rising part of the triangle modulation, the upsweep. Using the superheterodyne principle the received signal v_{Rx} and a portion of the transmitted signal v_{Tx} (called the local oscillator (LO)), are frequency mixed in an analog multiplier to get the intermediate frequency (IF) v_{Mixer}. The IF contains a target's beat frequency, which is proportional to the target's range. With the transmitted signal v_{Tx} and the received signal v_{Rx} as a function of time t,

$$v_{Tx}(t) = A_{Tx} \cos\Big(\omega_{Tx}(t)\ t\Big)$$
$$v_{Rx}(t) = A_{Rx} \cos\Big(\omega_{Tx}(t-\tau)\ t\Big)$$

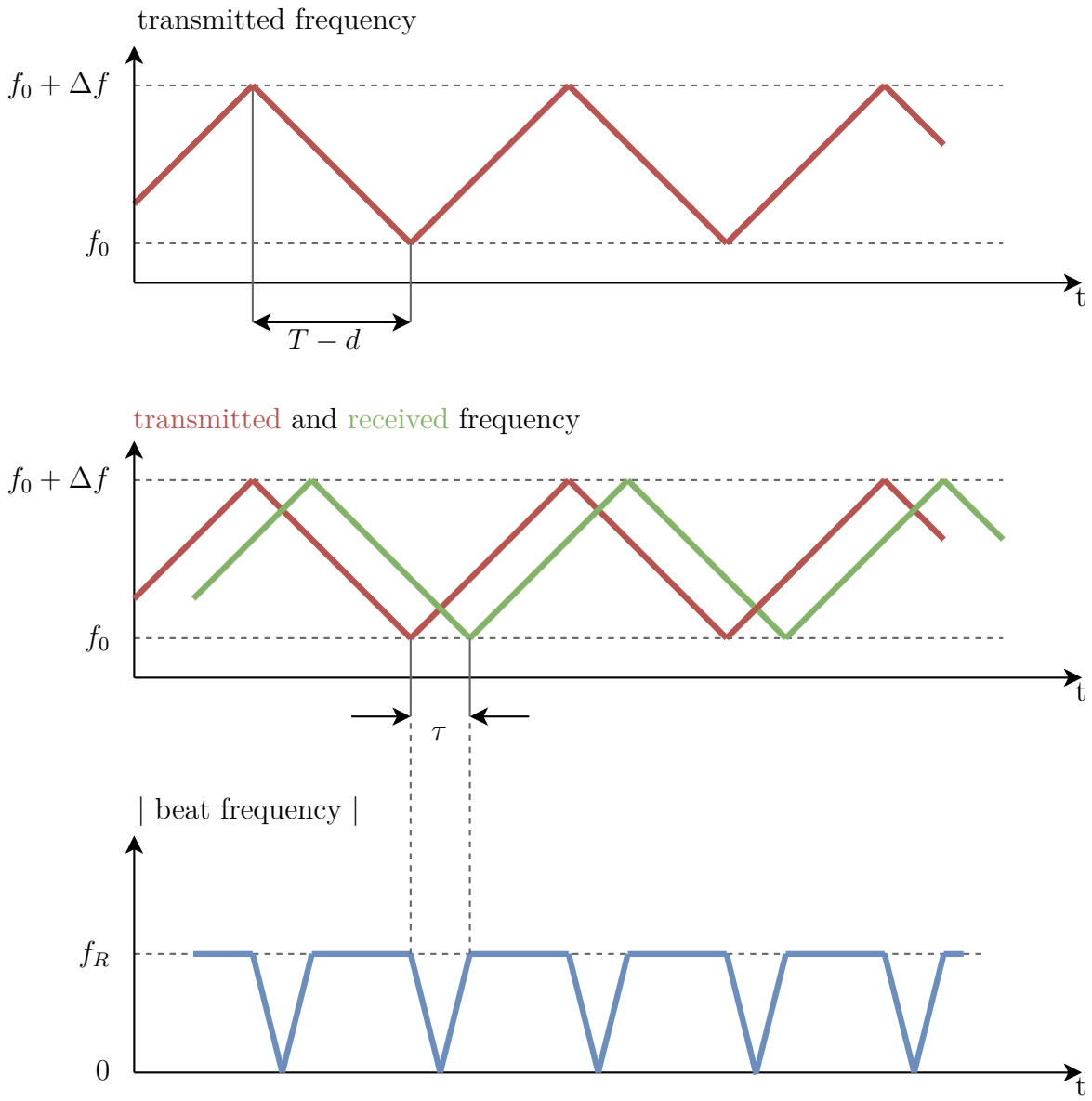


Figure 1.3: FMCW radars detect targets in the beat frequency, which is a frequency mix of the transmitted and received modulation. Adapted from [**Adams2012**] p. 57

where ω_{Tx} is the angular frequency of the transmitted signal,

$$\omega_{Tx}(t) = \underbrace{\omega_c}_{\text{Carrier frequency}} + \underbrace{\pi \frac{\Delta f}{T_d} t}_{\text{Upsweep modulation}}$$

the signal behind the frequency mixer v_{Mixer} can be calculated as

$$\begin{aligned} v_{Mixer}(t) &= v_{Tx}(t)\ v_{Rx}(t) \\ &= A_{Tx}A_{Rx}\ \cos(t\omega_{Tx}(t))t\ \cos(\omega_{Tx}(t-\tau)t) \end{aligned}$$

With the trigonometric identity

$$\cos A \cdot \cos B = \frac{\cos(A+B)\ +\ \cos(A-B)}{2}$$

v_{Mixer} can be written as

$$v_{Mixer}(t-\tau) = \frac{A_{Tx}A_{Rx}}{2}(B_1 + B_2)$$

where

$$\begin{aligned} B_1 &= \cos\Big[2\omega_{Tx}(t-\tau)t - \omega_{Tx}(\tau)\tau\Big] \\ &= \cos\left[2\left(\omega_c + \pi\frac{\Delta f}{T_d}(t-\tau)\right)t - \left(\omega_c - \pi\frac{\Delta f}{T_d}\tau\right)\tau\right] \\ B_2 &= \cos\left[2\left(\pi\frac{\Delta f}{T_d}t\right)\tau - \omega_{Tx}(\tau)\tau\right] \\ &= \cos\left[2\pi\left(\frac{\Delta f}{T_d}\tau\right)t - \left(\omega_c + \pi\frac{\Delta f}{T_d}\tau\right)\tau\right] \end{aligned}$$

Note that B_1 consists of high angular frequencies around the carrier frequency, from $f_0 = \frac{\omega_c}{2\pi}$ to $f_0 + \Delta f$. B_2 is a lower frequency (theoretically up to $2\pi\Delta f$ at $\tau = T_d$, but much lower in practice, as echos from targets this far away have very low intensity A_{Rx} and can't be detected) signal containing the beat frequency. The output of the low-pass filter intrinsic in the mixer stage will thus only consist of the beat frequency (plus noise of similar frequencies):

$$v_{beat}(t,\tau) = \frac{ATxA_{Rx}}{2}\cos\left[2\pi\left(\frac{\Delta f}{T_d}\tau\right)t - \left(\omega_c + \pi\frac{\Delta f}{T_d}\tau\right)\tau\right]$$

The term $\frac{\Delta f}{T_d}\tau$ in $v_{beat}(t,\tau)$ is known as the beat frequency f_b. For stationary targets, the range-specific frequency $f_R = f_b$. Knowing that the delay time τ depends on the speed of light c and the range R, the relationship between a target's f_R and range R can be given as

$$f_R = \frac{2R}{c}\frac{\Delta f}{T_d} \iff R = \frac{cT_d}{2\Delta f}f_R$$

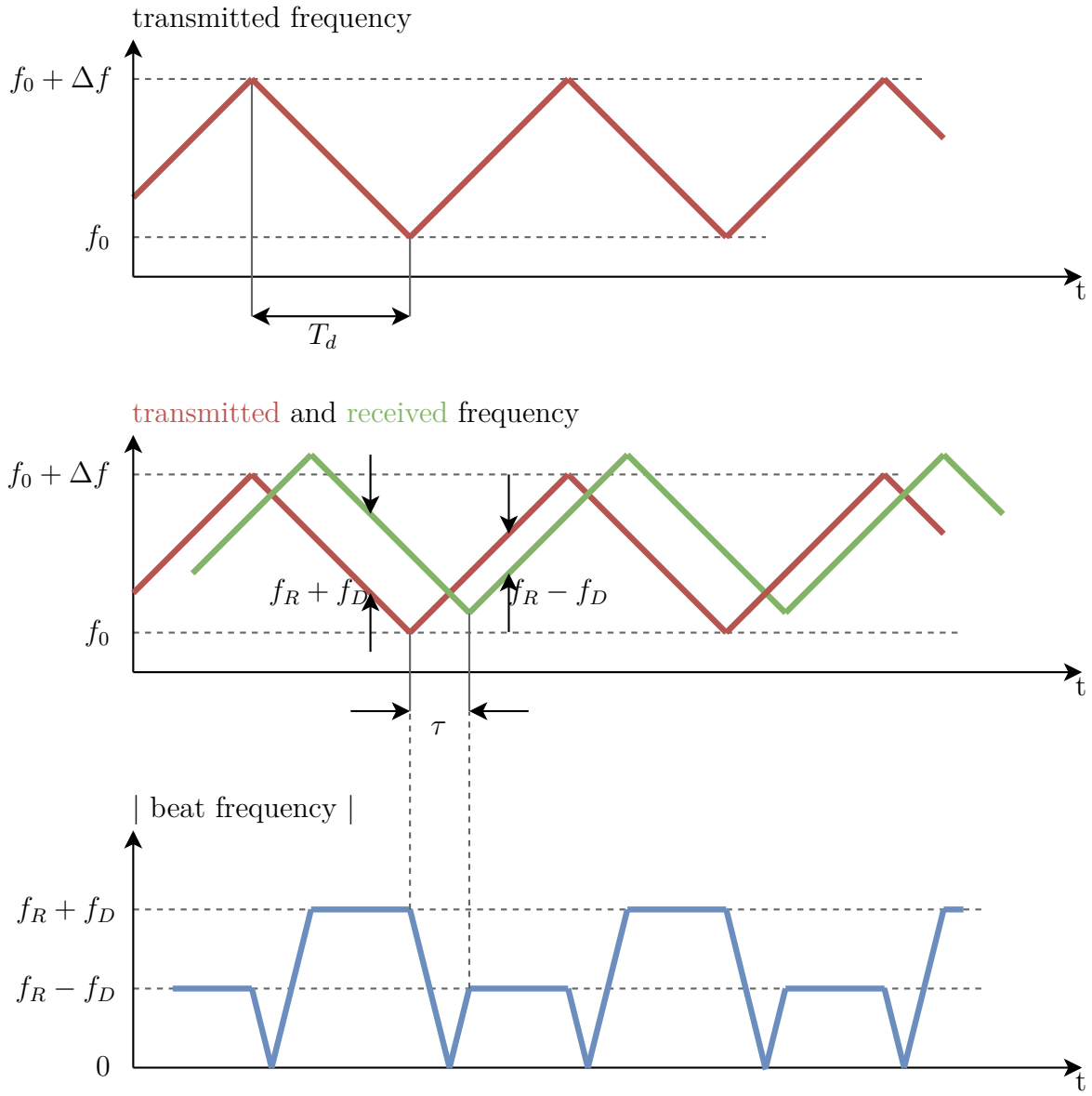


Figure 1.4: Target motion introduces a Doppler shift in the received signal, which changes the beat frequency during up- and down-sweep. Adapted from [**Adams2012**] p. 57

This also gives the range resolution dR,

$$dR = \frac{c}{2\Delta f}$$

A moving target however will introduce a Doppler shift f_D in the received signal v_{Rx}, which will shift the target's beat frequency f_b away from its range-specific frequency f_R. The direction of the frequency shift depends on the modulation: An up-sweep will have a corresponding

$$f_{b,up} = f_R - f_D$$

while the down-sweep has

$$f_{b,down} = f_R + f_D$$

The range-specific frequency and Doppler frequency can be extracted from the two beat frequencies by averaging them:

$$f_R = \frac{f_{b,down} + f_{b,up}}{2}$$
$$f_D = \frac{f_{b,down} - f_{b,up}}{2}$$

The benefit of the triangular sweep becomes clear here: with a sawtooth waveform, only f_b, up can be determined. A stationary target and a moving target a range and Doppler speed corresponding to the same resulting frequencies would not be distinguishable.

Of course, more than one target can be visible at a time. If multiple echos are received, as in figure **??**, the intermediate frequency will contain multiple frequencies. The beat signal will have more than one dominant frequency in its spectrum, with each one corresponding to a different target.

Figure **??** shows a real-world example of how the beat signal f_b (top subplot) and its frequency spectrum (bottom subplot) look like. At $t = T_d = 32ms$, a jump in the beat frequency is caused by the modulation change from upsweep to downsweep. In the frequency spectrum, three stationary targets are visible at ca. $3kHz$, $6kHz$, and $9kHz$. In this example, $T_d = 32ms$ and $\Delta f = 7GHz$, so the targets were at ranges of $0.5m$, $1m$, and $1.5m$. Note that the in-phase part S_I and quadrature part S_Q of the analytic signal are measured in separate channels to retain phase information. The Fourier transform in bottom subplot however shows the magnitude $\|S\|$ of the analytic signal[1)] $S = S_I + j\ S_Q$ with the imaginary unit j.

1.2.4 Direction of Arrival

A radar sensor with two or more receiving antennas which are separated by not more than half a wavelength can measure the Direction of arrival (DOA) of one or multiple targets.

[1)] A good explanation is available at [ping.se's I/Q Data for Dummies](http://whiteboard.ping.se/SDR/IQ)

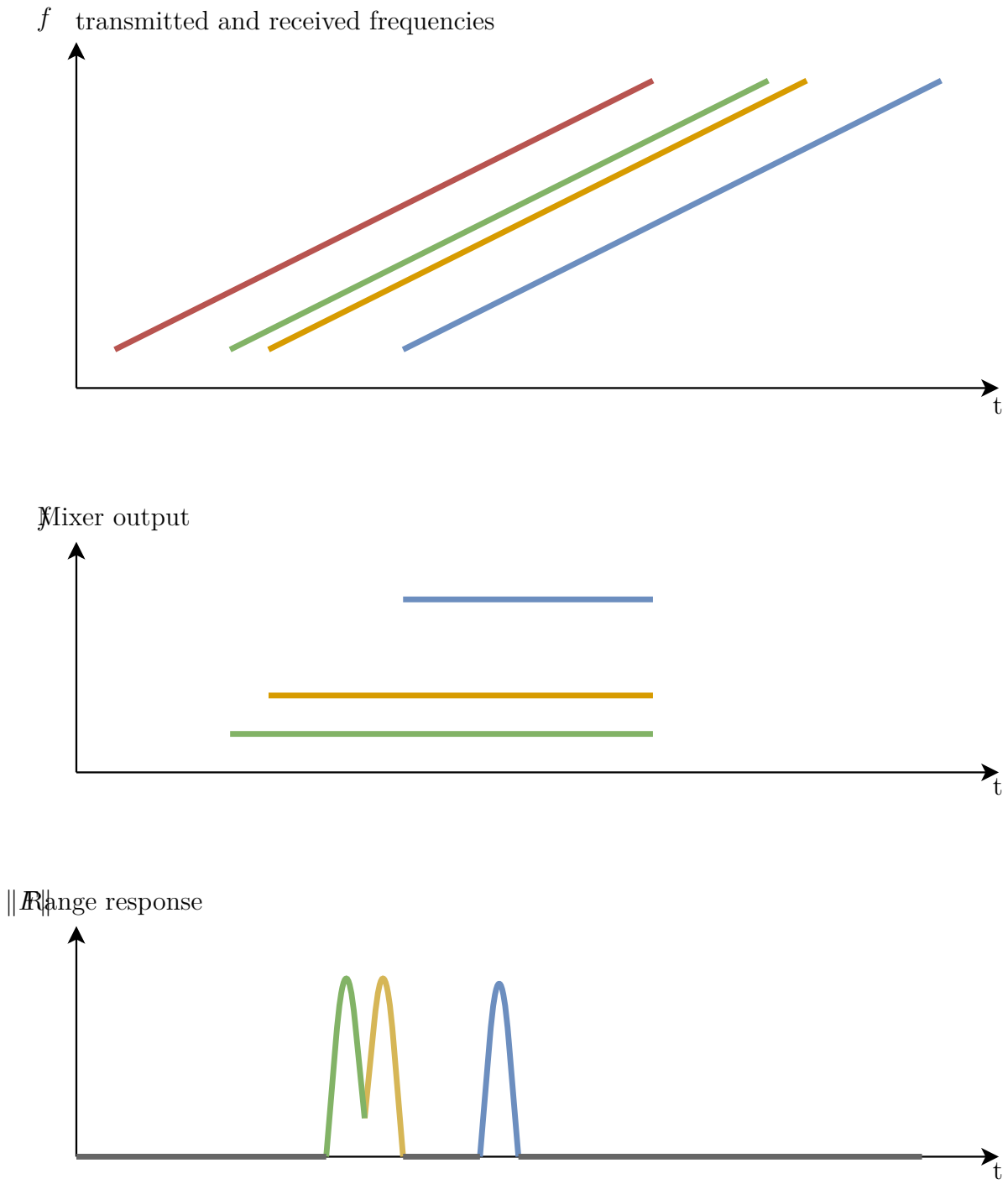


Figure 1.5: Each target contributes a frequency corresponding to its range in the beat spectrum.

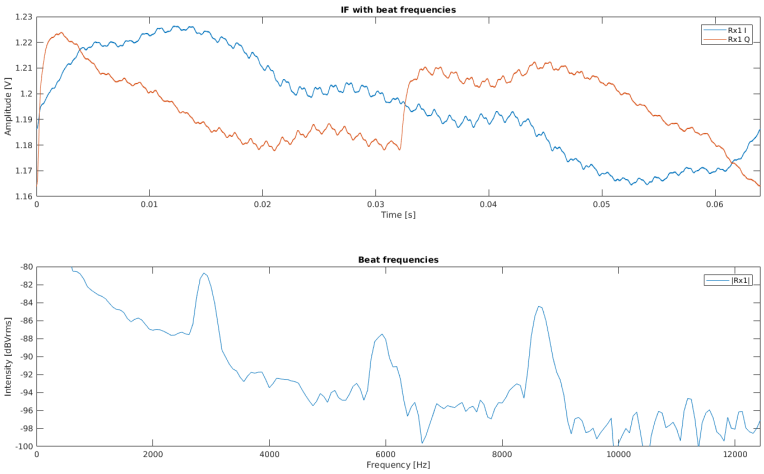


Figure 1.6: Beat signal and spectrum / range profile in a real-world measurement.

Because the echo from a target has to travel a slightly longer distance to antennas further away, a phase difference between the different antenna signals is measurable.

In the case of a pair of receiving antennas, the direction of arrival θ can be estimated [**VanZeijl2014**] as

$$\theta = sin^{-1}\left(\frac{\lambda\ \Delta\Phi}{2\pi\ d}\right)$$

with wavelength λ, antenna separation d and phase difference $\Delta\Phi$. $\Delta\Phi$ is the angle difference $\angle S_1 - \angle S_2$ of the complex analytic antenna signals $S = S_I + j\ S_Q$.

TODO [**Hacker2010**] [**Cho2017**]

1.2.5 Frequencies

Even though radar applications exist for many frequencies, only a few of them are OK to use for radiolocation and in home robots. The "Industrial, Scientific, Medical" (ISM) bands allows the unlicensed use of some frequencies for radiolocation, including center frequencies of $24.125GHz$, $61.25GHz$, $122.5GHz$ and $245GHz$. Applications must however accept harmful interference [**FCC2017**].

The $77GHz$ band is "restricted to vehicle-mounted field disturbance sensors used as vehicle radar systems." (FCC Part 15 §15.253(c)) - ETSI defines it as "Automatic Cruise Control 'long-range radar' operating at $77GHz$. This enables a vehicle to maintain a cruising distance from a vehicle in front." (EN 301 091). The German Bundesnetzagentur also declares it "Kraftfahrzeug-Kurzstreckenradar" (Vfg 66 / 2014).

A $24GHz$ center frequency is a safe bet. There are many radar systems available and it being an ISM band makes licensing much easier. The drawback is the very limited

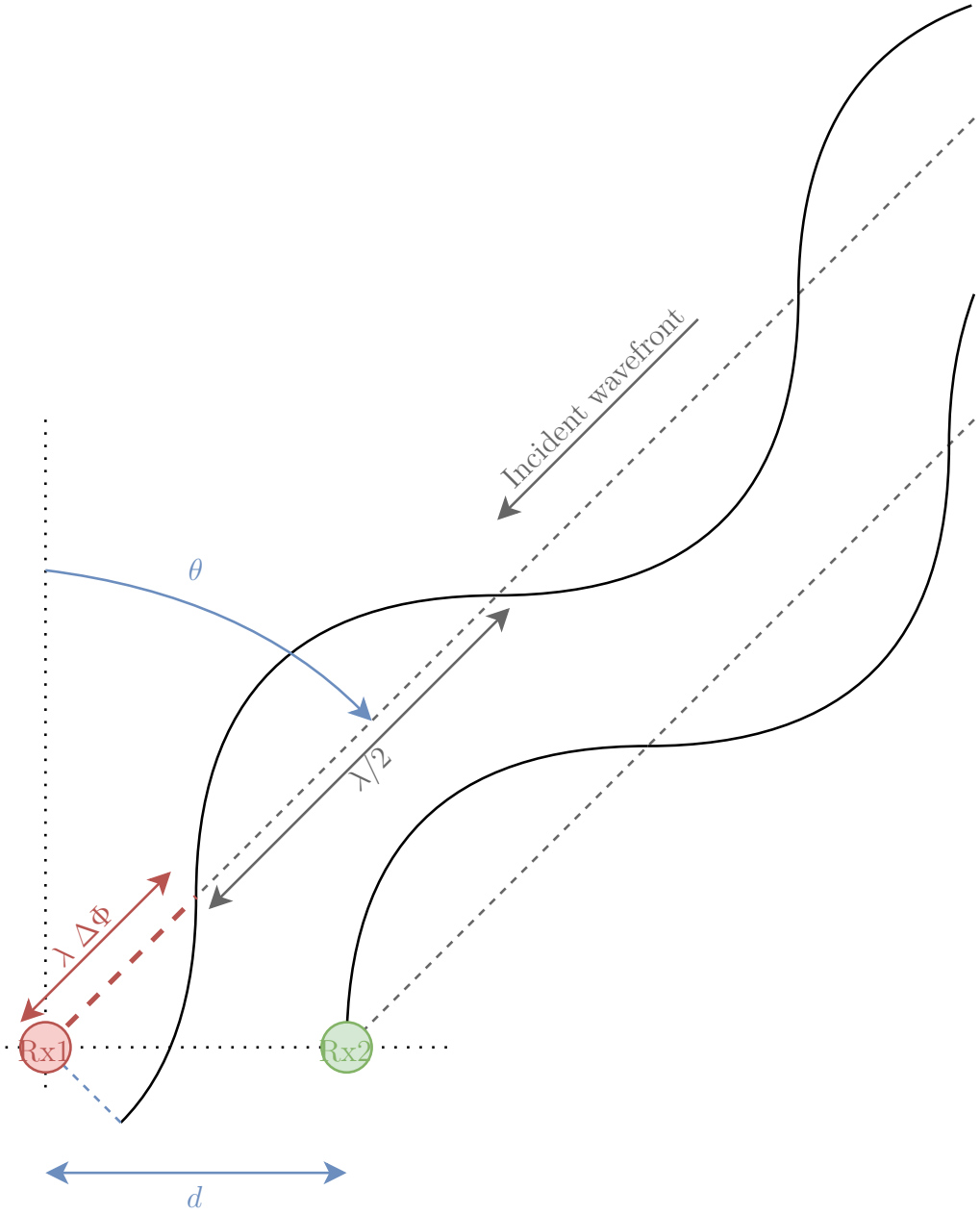


Figure 1.7: Direction of Arrival θ can be estimated from phase difference $\Delta\Phi$

maximum bandwidth of $250MHz$ in this band.

Some newer radars use the $60GHz$ ISM band. It allows a rather wide bandwidth of up to $9GHz$ in some regions (see figure ??). According to equation #REF, this gives a very good range resolution in the order of few cm. At these high frequencies, RF energy attenuation in material increases noticably [**FerrisJr.1998**]. The effect is that $60GHz$ waves are limited to short ranges of a few m and don't usually penetrate walls.

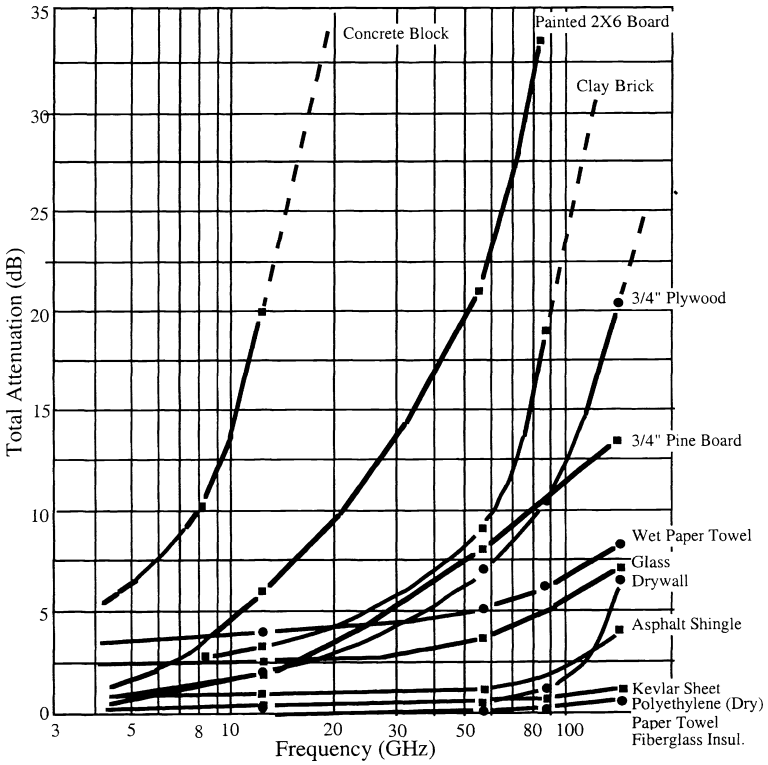


Figure 1.8: Totl attenuation of RF energy when transmitted through various materials as a function of frequency. Source: [**FerrisJr.1998**]

Atmospheric attenuation also limits long-range applications. In short range (a few m) it should however not present a problem.

Figure 1.9: Specific RF attenuation due to atmospheric gases. Source: [**ITU1997**]

The downside is that there are some other technologies using the same frequency bands, most notably 802.11ad a.k.a. WiGig[**AgilentTechnologies2013**]. The WiGig frequency allocations in figure ?? show in which regions the $60GHz$ band is available (also for radar).

Figure 1.10: WiGig Channel Plan and Frequency Allocations by Region. Source: [**AgilentTechnologies2013**]

1.3 Overview of Radar Research

Radar is used and researched since the 1940s. While it was historically only used to detect aircraft and ships, it is an active research domain in many fields today. Identification and localization of vessels is of course still an important application in both civil and military sectors. There is a great amount of research going into synthetic aperture radar, in terrestrial imaging, but also general and concealed imaging. Another area of research is radar antenna technology and quasi-optics, which aims to find design improvements and more adapted antennas for the manifold applications. Radar is used in human presence detection and monitoring, including heartbeat detection. A new and very promising discipline is radar-based gesture recognition, which enables innovative human machine interaction applications. Indoor communication and localization with radar beacons is another interesting and upcoming technology. The radar-related research area that is most relevant for mobile robots is radar-based slam.

1.4 Existing radar-based solutions for map building

1.4.1 SAR

In 1950 Doppler frequency analysis was found to improve image resolution of side-looking radar, which led to the development of the synthetic aperture radar (SAR) technique. SAR uses azimuth (along-track) motion to synthesize an aperture that is longer than the physical size of the radar antenna [**Wang2008**]. The three major configurations are stripmap SAR, scan SAR, and spotlight SAR. Current SAR systems can operate in either mode by dividing their planar antenna into sub-apertures, whose phase and amplitude are controlled by the individual few hundred transmit/receive modules [**Moreira2013**].

Figure 1.11: Illustration of different SAR operation modes which are used to increase the swath width (ScanSAR) or improve azimuth resolution (Spotlight) compared to Stripmap mode. (a) Stripmap. (b) ScanSAR. (c) Spotlight. Source: [**Moreira2013**]

Radar echo data is sampled in both fast-time and slow-time, with fast-time meaning the range scan dimension (fast, because the EM waves travel at very high speed, c) and slow time denoting the azimuth or along-track dimension (slow, because movement velocity will be $\ll c$). This raw data does not give any useful information and needs to be signal-processed first. Because SAR systems typically use pulse-compressed radars, each range line needs to be convoluted with the complex conjugate of the transmitted chirp's spectrum to obtain the range-compressed data[2)]. In a second step, azimuth compression

[2)] For the FMCW system used in the later parts of this thesis this is not necessary - the FMCW beat frequency spectrum is equivalent to the range-compressed data

takes place by convolving the signal in slow-time with the complex conjugate of the expected azimuth-response from a target.

Figure 1.12: Summary of SAR processing steps where the range compressed data result from a convolution of the raw data with the range reference function. In a second step the azimuth compression is performed through a convolution with the azimuth reference function, which changes from near to far range. Source: [**Moreira2013**]

An elemental scatterer at range $R(t)$ will return an echo $s_a(t)$ over time t:

$$s_a(t) = P_r\sqrt{\sigma}\exp(j\varphi_s)\exp(j\underbrace{\frac{-4\pi}{\lambda}R(t)}_{\text{az. phase var. } \omega_D})$$

where P_r is the echo power of the received target, accounting for dependencies like transmit power and path loss, σ is the target's RCS, imaginary unit j, φ_s the scattering phase, and $\frac{4\pi}{\lambda}R(t)$ the azimuth phase variation due to changing distance [**Cumming2004**]. The target's range $R(t)$ is described by the range at closest approach R_0 and the radar's (constant) movement speed v_R:

$$R(t) = \sqrt{R_0^2 + (v_R t)^2} \approx R_0 + \frac{(v_R t)^2}{2R_0} \text{ for } \frac{v_R t}{R_0} \ll 1$$

Substituting #REF into the azimuth phase in #REF and derivating with respect to time yields the azimuth frequency f_D

$$\begin{aligned} f_D &= \underbrace{\frac{1}{2\pi}}_{\omega_D = 2\pi f_D} \frac{\partial}{\partial t}\frac{-4\pi}{\lambda}\underbrace{\left(R_0 + \frac{(v_R t)^2}{2R_0}\right)}_{R(t)} \\ &= -\frac{2v_R^2}{\lambda R_0}t \end{aligned}$$

The azimuth frequency varies linearly with time t and is inversely proportional to the closest approach (slant range) R_0, hence the azimuth reference function depends on geometry and is adapted to range. Because of the frequency-shifting effect it is analogous to and also called the Doppler frequency.

The most challenging aspect of SAR is the correction of range cell migration induced defocusing. Range cell migration is visible in figure ??'s curvature of range compressed data. It occurs when a point target's echo energy is distributed over several range cells, causing azimuth defocusing. This effect is range-variant, as the curvature depends on R_0. Hence a non-stationary two-dimensional reference function is necessary. There are several approaches in tackling this, including the omega-k / wavenumber processor, range-Doppler, and chirp scaling algorithms [**Moreira2013**].

1.4.2 Scanning radar

TODO

Small ground robots

- Martin Adams
- Henrik Forsten
- Gregory Charvat
- Bat type radar

1.4.3 Radar slam

TODO

Radar Slam

- Martin Adams
- K2Pi

2 Novel Approach: Reprojection Mapping

2.1 Idea

With the Reprojection method, a radar echo's source can be determined without scanning the radar sensor. The distance to a target is already available in the radar data, so only the direction is needed to update a map. Under certain circumstances, this direction can also be extracted from the sensor's data.

The prerequisite to the method is that the sensor has to move with a known speed v_R through an otherwise static environment. Caused by the radar's motion, the distance to all visible targets changes, which causes a Doppler speed v_D for every target point.

2.1.1 Geometry for the side-facing case

If the radar is moving directly towards a target, the target's Doppler speed v_D will be equal to radar speed v_R (target A in figure ??). If the radar passes the target on the side, so that the distance to the target stops decreasing and begins to grow, $v_D = 0$ (target C in figure). If a target has an absolute Doppler speed $\|v_D\|$ between 0 and v_R (target B in figure), it is seen from the radar under an angle α, such that

$$v_D = v_R cos(\alpha)$$

Figure 2.1: Reprojection geometry for the side-facing case

Positive values for α indicate targets on the left side of the robot, while negative values for α indicate right hand side targets. The caveat is that from v_D only the absolute value of angle α is known. This is OK if the radar is mounted side-facing, such that only targets on one side of the motion path are visible. Like this, the angle ambiguity is resolved, because only one side is visible.

2.1.2 Geometry for the General case

In the general case such as with a forward facing radar, or with radar antennas with angle sensitivities that allow a field of view of over 180°, the angle ambiguity must be resolved differently.

One solution to resolving the angle ambiguity is to use direction of arrival (DOA) information from a multistatic radar. If the radar is facing straight forward, a target's DOA will be lower or higher, depending on the antenna configuration and the side the target is being passed on. If the radar is not facing straight but still has parts of both sides of the motion paths visible in it's FOV the DOA gradient should be used instead of its value: If a target's DOA is gradually shifting towards left (i.e. lower or higher, depending on the system), it will pass the radar on the left side. If it is gradually shifting towards right, it will pass on the right side.

2.1.3 Reprojection Method

TODO

When a target's range and angle are known, it can be mapped in relation to the radar's position.

This allows intelligent path planning and obstacle avoidance for mobile robots.

2.1.4 Peak Gradient algorithm

TODO

One pillar of the Reprojection Method is the Doppler speed of static targets. If the Doppler speed is not precisely measured, the reprojection angle α is imprecise or noisy, which leads to smeared-out targets or even false positive detections on the map.

With FMCW radar, a target's range and Doppler speed can be simultaneously registered.

2.1.5 Limitations

TODO

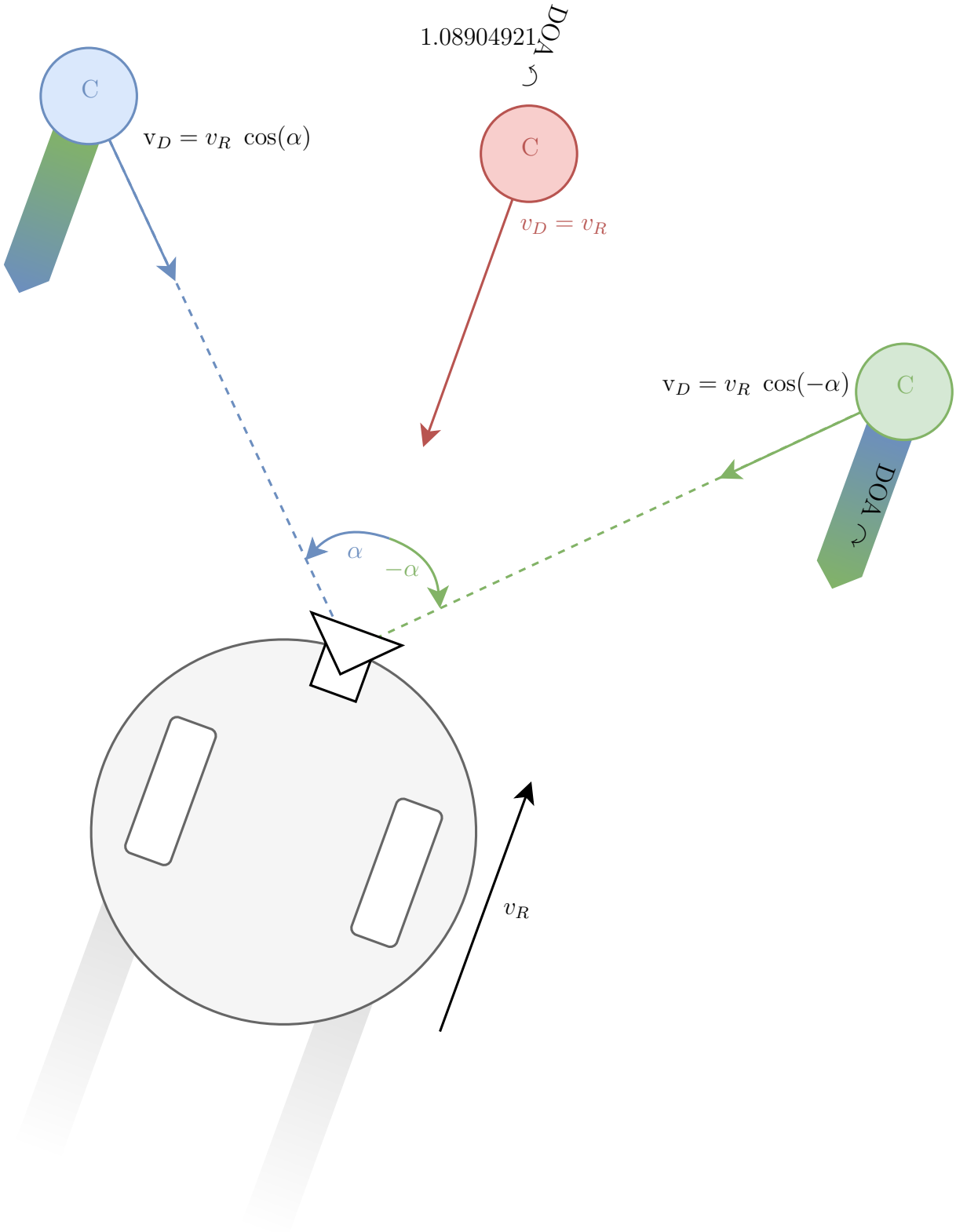


Figure 2.2: Reprojection geometry for the general 2D case

3 Implementation

3.1 Implementation Platform

The validity and performance of the concept is put to test with a proof of concept implementation. The reprojection method requires that the radar position and velocity is known at all times. While there was no precise source of ground truth data (like an IR ceiling camera system) was easily available, the Kobuki robot platform provides a fairly good odometry system and also allows to move the radar in a controlled way.

3.1.1 Kobuki

TODO

Good but not perfect odometry Ros Arm Limited performance Enough battery capacity Omniradar FMCW Radar development kit

Figure 3.1: Yujin's iClebo Kobuki robot platform. Source: [**DesignK2013**]

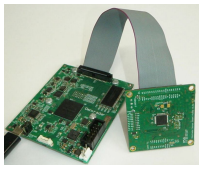
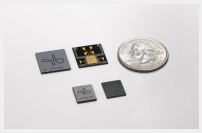
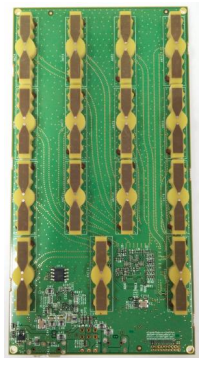
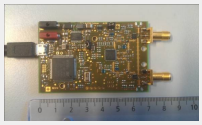
ROS integration

Odroid

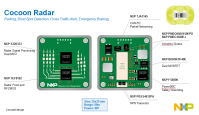
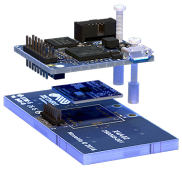
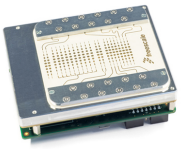
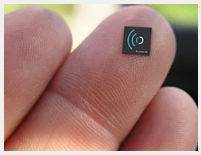
3.1.2 Radar sensor

Devkit list

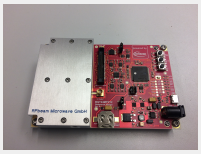
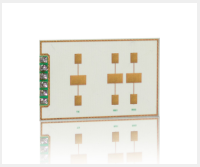
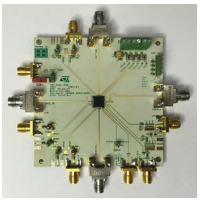
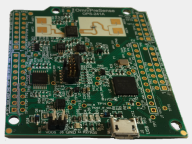
There are quite a few short-range UWB FMCW radar modules. The following table compares some promising solutions.

Product	Note	f_C, Δf	Antennas	DK Price	Picture
Omniradar RIC60A	High bandwidth. Presentation at SoC 2015[**Brouwer2015**]	60 GHz, 7 GHz	On-chip, 1 Tx, 2 Rx	$4000	
Google / Infineon Soli	Expected 2018. Sub-millimeter accuracy, running at over 10,000 frames per second [**Lien2016**]	60 GHz, 7 GHz	In-package, 2 Tx, 4 Rx	?	
Walabot Pro	3D configuration. Slow update rate	6.8 GHz, 7 GHz	On-board, 9 Tx, 9 Rx	$599	
Bosch Prototype	Prototype for In-wall pipe detection	5.15 GHz, 6.7 GHz	External, 2 Tx/Rx	$0	
Silicon Radar SiRad Simple	Has WiFi	122 GHz, 6.4 GHz	On-chip, 1 Tx, 1 Rx	?	

Continued on next page

Product	Note	f_C, Δf	Antennas	DK Price	Picture
Anokiwave AWMF-0117		12.5 GHz, 4.5 GHz	On-chip, 1 Tx/Rx	?	
NXP Cocoon Radar	Relatively small board. Presentation at FTF 2016[**Reuter2016**]	77 GHz, 4 GHz	On-board, 3 Tx, 4 Rx	?	
TimeDomain P440	Can operate as multistatic radar or UWB communication node	4 GHz, 1.7 GHz	External, 2 Tx/Rx	$5000	
Novelda Xethru X4M03		8 GHz, 1.5 GHz	On-board, 1 Tx/Rx	$399	
RFbeam MR2001_RD		77 GHz, 1 GHz	On-board, 4 Tx, 6 Rx	?	
Inras 77Ghz Radarbook	Configurable FPGA processing chain	77 GHz, 1 GHz	On-board, 4 Tx, 8 Rx	$7300	
Acconeer A1	Sub-mm accuracy	60 GHz, ?	On-chip, ?	?	
Inras 24Ghz Radarbook		24 GHz, 250 MHz	On-board, 4 Tx, 4 Rx	$7300	

Continued on next page

Product	Note	f_C, Δf	Antennas	DK Price	Picture
Infineon BGT24-RFB2412-EVAL		24 GHz, 250 MHz	On-board, 1 Tx, 2 Rx	$1333	
IMST DK-sR-1200e		24 GHz, 250 MHz	On-board, 1 Tx, 2 Rx	$3333	
InnoSenT IVS-565		24 GHz, 250 MHz	On-board, 1 Tx, 2 Rx	?	
ST EVB-STradA431	SMA connectors for internal signals	24 GHz, 250 MHz	External, 1 Tx, 3 Rx	?	
OmniPreSense OPS241-A	Arduino shield with BGT24LTR11	24 GHz, 80 MHz	On-board, 1 Tx/Rx	$169	

The most suitable modules are the ones with the highest bandwidth as they give the highest resolution. Other beneficial properties are high update rate, and higher number of antennas. A high degree of configurability also helps during development. In the end, the Walabot Pro, Omniradar RIC60A and a proprietary Bosch prototype were available to be tested.

Bosch Radar

The Bosch radar presented some challenges under a Linux environment. After its Matlab driver was patched for cross-platform compatibility, it turned out that the on-board MCU's firmware had an incompatible protocol format. A newer version of the prototype did work under Windows, but by the time the board arrived, the decision to focus on Omniradar was made.

Walabot

Vayyar is an Israeli startup that was founded in 2011. Coming from a medical background, they moved away from use cases such as breast cancer detection towards general 3D imaging in the consumer market with their Walabot sensor.

Vayyar's Walabot Pro sensor uses an 18-antenna MIMO array for 3D radar imaging. Vayyar is very quiet about the technology and algorithms used in their product and even the nature of the data that the sensor sends. In their Python API documentation they showcase the modes of operation: 3d imaging, 2d imaging, object tracking, pipe detection and raw data.

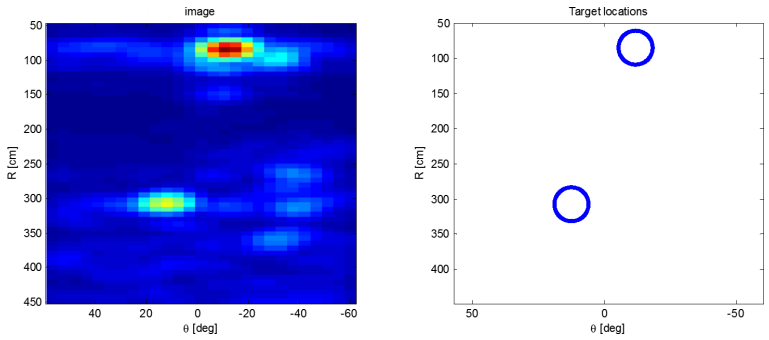


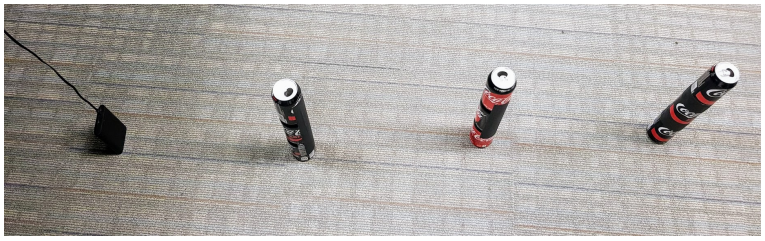
Figure 3.2: Vayyar's Walabot Pro sensor is claimed to have Target localization and tracking. Source: https://api.walabot.com/_features.html#_examples

The catch is however that it is almost impossible to do imaging without background subtraction, which they do in all their examples. This works well in scenarios where the sensor is at a fixed position or if the region of interest is very small, like in the pipe detection scenario. However, In the case of a robot-mounted sensor this does not work so well.

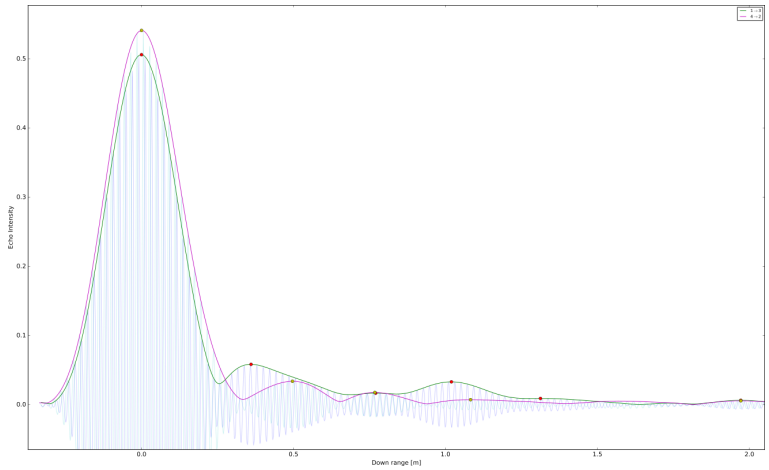
At the time of writing, the only interesting published project is a fall detection scenario [**Haider2017**] by Haider, in which people can be localized at intersections of vertically and horizontally oriented heatmaps.

Static range test Figure #REF shows the signal from two Walabot antenna pairs as it records the scene in image #REF with can stacks at $0.5m$ $1m$ and $1.5m$. The signal is very stable over time and shows next to no noise. Unfortunately however it doesn't seem like the radar sensor can detect the metal cans at all. The high frequency signal is the raw signal as reported by the Walabot sensor. It is however hardly believable that it was measured like this, as the visible base frequency of the signal is around 7GHz. It was not possible to get any useful information from Vayyar's technical support regarding this. The envelope signal is a more interesting data source. It can easily be obtained from the analytic signal, using `scipy.signal`'s Hilbert transform. Another problem with the data is that the peaks of the envelope jitter in range. This can be fixed by combination with another oddity: The last 180 samples rise very strongly in magnitude. If they are cut and

prepended to the first sample, they match up perfectly. Peaks can then be detected in the signal (represented by the dots figure #REF) and the range set to zero at the first peak. This eliminates the range jitter completely. The reason is that the first peak is caused by transmit antenna crosstalk and can thus be used as a timing reference point.



(a) Setup with three towers of cans in front of the Walabot sensor



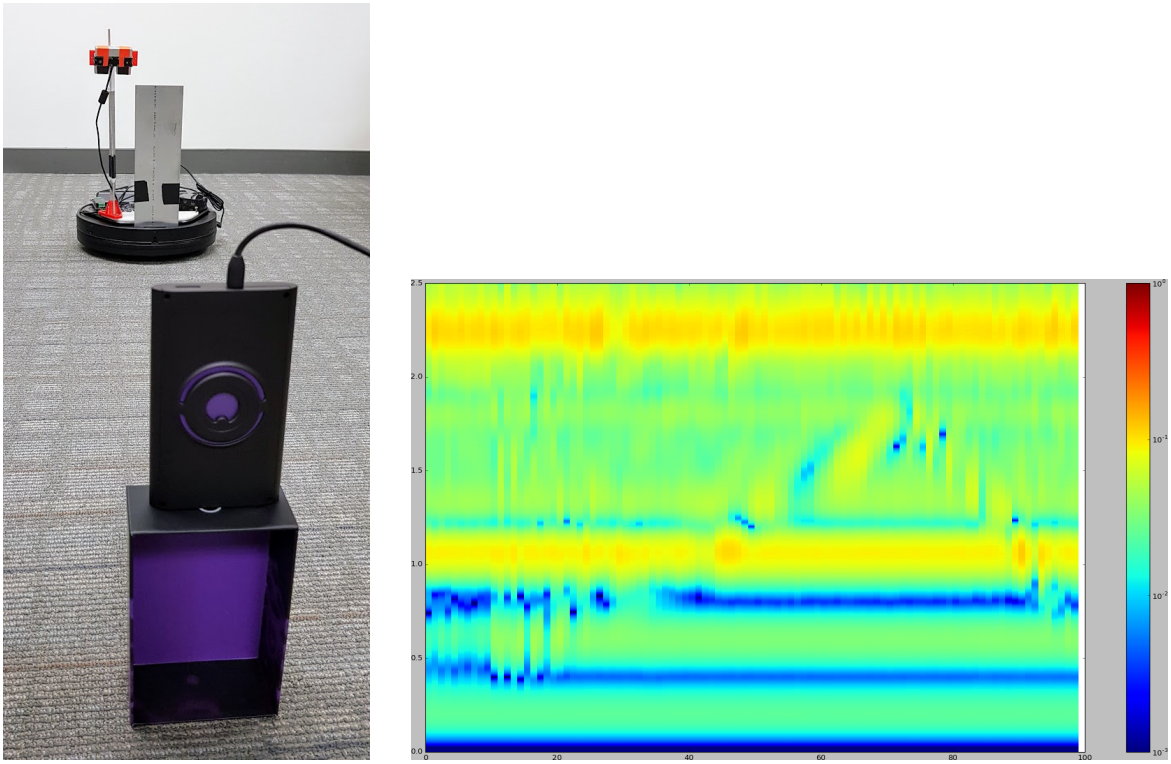
(b) Range profile measurement with two antenna pairs. After the transmit peak at $t = 0$, three peaks corresponding to the three can targets are vaguely perceptible.

Figure 3.3: Static range test

Dynamic range test Waving hands in front of the sensor did give a change in signal, but it was difficult to interpret the data conclusively. To objectively test the sensor's response, an aluminum plate that gave a strong echo signal was taped to the Kobuki robot. The robot was then driven with a constant speed away from the sensor and then towards the sensor as pictured in **??**.

The sensor was sampled at a constant frequency in raw data mode. The analytic signal magnitude of the range scans is stacked at the right end of the matrix displayed in figure **??**.

The figure shows that the Walabot has problems with what looks like standing waves. The great amount of background signal is also visible. Because of its static nature, this can easily removed for a fixed radar. As some Walabot reviewers have noticed [**Valens2016**] this background signal changes heavily and seemingly random when the sensor is moved. This makes the signal processing very difficult.



(a) Setup with a reflective metal plate mounted on the robot.

(b) Range profile (*y-axis*) over time (*x-axis*) while the robot was moved at constant speed away and towards the Walabot sensor.

Figure 3.4: Dynamic range test

Walabot advertises object detection capabilities. The catch with this mode is however that the number of objects to be detected must be configured first.

3.2 Omniradar

With its good range resolution and a good idea of its capabilities from [**Ernst2016**], this sensor promised good results. As it was chosen as a basis for the proof of concept implementation, it receives a more detailed look.

Founded in 2010, Omniradar is a Dutch startup that claims to be the first to integrate a complete 60 GHz radar including antennas and analog to digital conversion in one chip.

With the RIC60-A they offer a Radar Development Kit (RDK) that gives 7GHz of bandwidth on two receiving antennas. An Altera Cyclone IV FPGA handles the signal acquisition and communication. Figure #REF shows the radar IC and how the three antennas are integrated in silicon.

Figure 3.5: Decapped Omniradar IC; five cent coin as size reference; and antenna directionality pattern of Omniradar's RIC60-A. Source: [**Brouwer2015**] p.9

The radar sensor's beam is fan shaped, which means it is fairly sensitive over a wide angle in azimuth direction, but relatively focused in elevation. This makes it a very good candidate for the radar reprojection, as targets can be seen from the robot in a wide field of view, but floor and ceiling reflections are kept at a minimum. Of course the sensor can also be rotated. Omniradar also supplies a horn-like extension for the sensor board, which forms the radar sensitivity into a pencil-shaped beam that is very focused at a narrow field of view.

3.2.1 Radar mount

A 3D-printed part makes sure that the radar sensor is firmly mounted on the robot as it explores its environment. The part was designed in OpenSCAD and printed on a Dremel 3D printer. The bottom mount hole positions were extracted from the mechanical drawings of the Kobuki Base [**YujinRobot2012**]; the side holes from the Altium layer document of the version 3.2 Omniradar interface board [**Omniradar2014**].

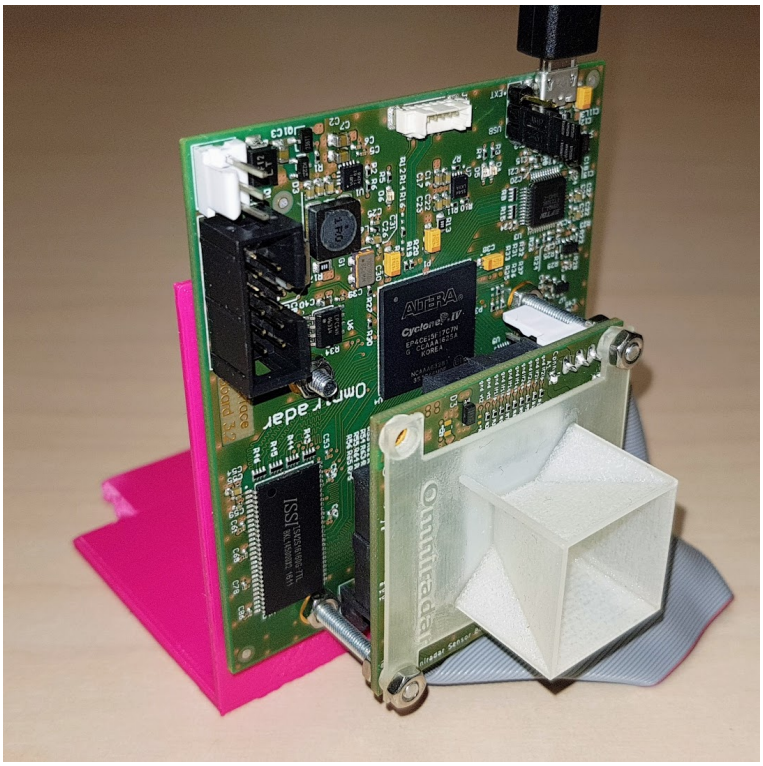
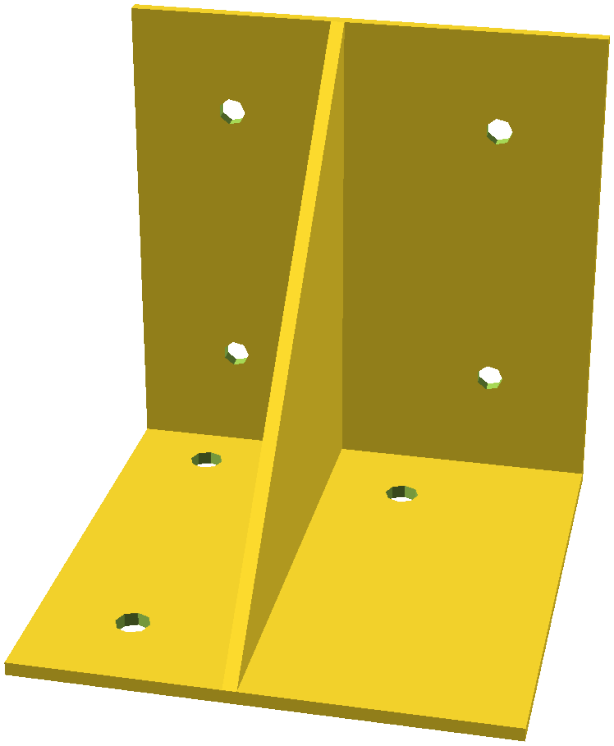


Figure 3.6: Omniradar's RIC60-A with horn antenna extension, attached to the 3D-printed Kobuki mount



When rotated to face to the left side of the robot, the RPLidar mount was in the way, so parts of the print had to be clipped off.

Two screws hold the Omniradar interface board to the mount. The other two mounting holes in the interface board hold the Omniradar sensor board. The horn extension can be affixed to these screws as well. If a different sensor orientation is necessary, the sensor board can be rotated by 90 degrees, thanks to the symmetric layout.

3.2.2 Doppler sensitivity

The Omniradar FMCW radar is not sensitive enough to use the Doppler speed directly. The following example illustrates this.

The RIC60A has a sensitivity of $400\frac{Hz}{m/s}$. A target with a doppler speed of $0.02m/s$ (A low speed at which the Kobuki robot still moves continuously and without jerking) will cause a frequency spike with a shift of $8Hz$ in the FMCW beat frequency.

The speed resolution capability is inversely proportional to the measurement or acquisition time. A 10 ms long acquisition gives a 100 Hz frequency resolution, or a speed resolution of 0.9 km/h (or 0.25 m/s).

Sampling frequency $F_s = 25MHz$ and RIC60A Doppler sensitivity, $S_D = 400\frac{Hz}{m/s}$ are constant values of the Omniradar RDK.

Given a chirp duration of $T_{chirp} = 2.5ms$, we get $N_s = t_{chirp}F_s = 62500$ Samples, $N_r = \lfloor \frac{N_s}{2} \rfloor = 31250$ Samples per up/downsweep, $dF = \frac{F_s}{N_r - 1} = 800Hz$ FFT frequency bin width

and hence a Doppler resolution of $\frac{dF}{S_D} = 2m/s$.

Even with subsample peak interpolation the accuracy will not be very good and targets will be reprojected at imprecise angles.

It would be possible to use higher precision equipment. But another solution is to track the movement of target peaks in the range profile, using the Peak Gradient algorithm.

3.2.3 Optimal chirp time configuration

The chirp duration t_{chirp} is configurable and has an effect on how the raw range profile data will look like.

Very short durations ($< 2ms$) incur a considerable processing overhead to acquisition time ratio and have a very low SNR. **Short durations** ($< 5ms$) have acceptable SNR, and are more efficient with respect to overhead. **Long durations** ($> 15ms$) have good SNR, and don't create a lot of overhead. However at higher robot speeds, target peaks get blurred over several range bins as they move in range. Less intense target echos are more difficult to detect then. **Very long durations** ($> 20ms$) required the Omniradar driver to be patched on Linux so as not to freeze when chirps longer than $20ms$ are requested. Even with the patched driver the RDK's FPGA firmware is not very reliable at sending large volumes of data at once and corrupts packet headers or aborts transmissions intermittently.

The optimal range was empirically found to lie between $2ms$ and $10ms$.

TODO wording here

Figure #REF shows the chirp efficiency $\eta = \frac{n_{chirp}\ t_{chirp}}{t_{msg}}$, with number of consecutive sweeps n_{chirp} (two in the graph's data source), chirp length t_{chirp}, and t_{msg} the time since last radar message.

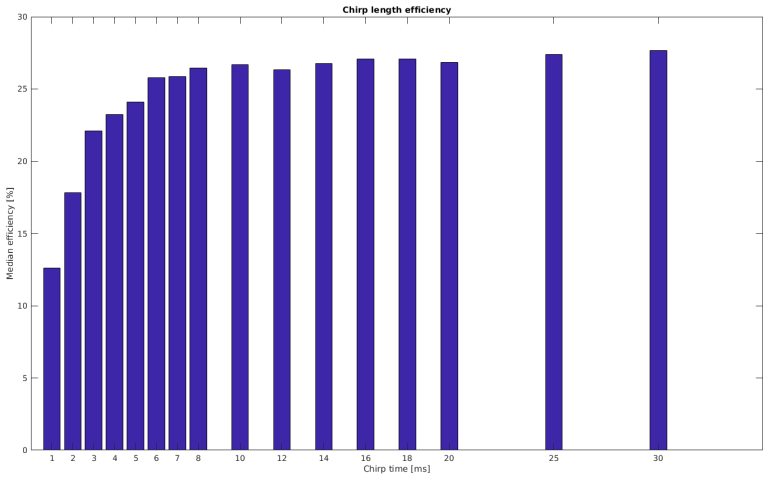


Figure 3.7: Chirp efficiency η for various chirp lengths

The chirp length has an effect on accuracy and resolution. Figure #REF shows how short chirp times

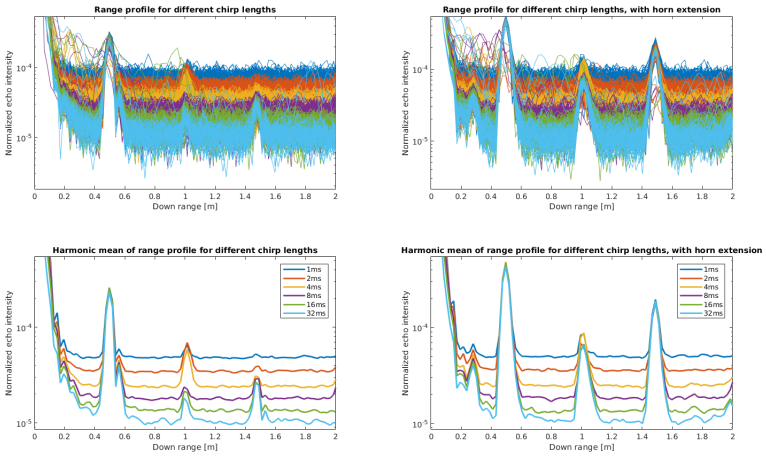


Figure 3.8: Effect of chirp length on SNR; with and without Horn extension

3.3 Omniradar ROS driver

The Omniradar RIC60A comes with a precompiled Matlab MEX driver library. This works well in a Windows OS on x86-based computers with a Matlab installation. An early goal was to have the robot carry the radar module around wireless. One option would have been to mount a Windows laptop on the Kobuki. However, Omniradar was kind enough to provide the author with the driver sources under an NDA agreement. This allowed recompiling the MEX driver for Linux systems. There were some issues with FTDI's D2XX serial communication library that had to be fixed for Linux systems. One challenge with the D2XX driver was that Ubuntu automatically loads the regular FTDI serial IO driver, `ftdi_sio`. This could be solved by unbinding the driver in a set of udev rules [ref]. Another issue was that due to a bug in the D2XX implementation, the Omniradar driver would freeze when more than 2MB of data (equivalent to a 20ms FMCW chirp) were requested. After a lot of debugging, this could be solved by never requesting a bigger amount of data than was already available in the D2XX buffer.

3.3.1 C++ bindings and library

Since Matlab does not run on Arm arch processors such as the Odroid on the Kobuki robot, a new set of platform-independent C++ bindings was added to the driver. The C++ bindings serve the same purpose as the Matlab bindings.

To include the library, some files need to be installed or pointed to by the binary that needs to use it.

File	Default install destination	Purpose
omniradar.h	/usr/local/include	Library header file
libomniradar.so	/usr/local/lib	dynamically linked shared object
51-omniradar.rules, 52-omniradar.rules	/etc/udev/rules.d/	Udev rules to unbind ftdi_sio

If the driver source is available, installing the files can be accomplished from the source directory with

```
mkdir build && cd build
cmake -DCPP_BINDINGS=ON -DMATLAB_BINDINGS=OFF ..
make
sudo make install
```

The driver can then be included in an application. Note that since it is dynamically linked, the `ftd2xx`, `pthread` and `dl` libraries dependencies also need to be linked. In a Catkin [1] CMakeLists.txt this would look like

```
target_link_libraries(
  ${PROJECT_NAME}_node
  omniradar
  ftd2xx
  pthread
  dl
  ${catkin_LIBRARIES}
)
```

The C++ library offers the same functions as Omniradar's Matlab driver, with two differences. The optional device index number is 0-based instead of 1-based and the AquireEcho functions return (a shared pointer to) packed data instead of unpacked data for performance reasons. With the

```
static std::shared_ptr< std::vector<std::vector<uint8_t> > > demultiplex(std::vecto
```

function, the library offers an easy way to demultiplex the packed data array into a (shared

[1] Catkin is the CMake-based ROS build system

pointer to a) vector of four vectors - one for each echo signal of the I/Q channel of left and right receiving antenna.

Useage is simple as the driver follows the RAII principle. Allocation of an object of type `Omniradar` causes the library to fully initialize the RIC60A RDK. After that, the configuration string and the VCO tuning curve should be set. Omniradar provides some Matlab example code that measures the sensor's VCO tuning curve. The easiest way to bring this tuning curve into the C++ domain is to print[2)] it as C style array, e.g. with

```
['#pragma once' 10 'std::vector<double>' 10 'vco_tune {', sprintf('%.100g, ', VCOtune),
```

and then saving the resulting string as `vco_tune.h` include file.

This setup allowed the development of a ROS node that handles communication with the radar module on the Linux/Arm based Kobuki robot.

3.3.2 ROS node

A new `omniradar` ROS package was developed to support the use of the Omniradar sensor within the ROS environment. A set of `roslaunch` launchfiles comes with the package that support the startup of the Kobuki robot in teleoperation mode, optionally together with lidar-based Cartographer slam, AMCL localization and an Astra Orbbec RGBD camera. A simple version to start the only the node to get radar data would be:

```
<launch>
  <node pkg="omniradar" type="omniradar_node" name="omniradar_node" output="screen">
    <param name="n_sweeps" value="1" />
    <param name="t_sweep" value="5" />
  </node>
</launch>
```

The node sends out the ROS topic `/omniradar_node/radar_raw` of the custom `RadarEcho` type, which is defined in RadarEcho.msg as

```
Header header
string ric_config
uint8 n_sweeps
float64 t_sweep
uint32[] packed_echo
```

Early versions of the driver unpacked the radar echo bitstream inside the node and were able to use standard ROS message types like the `std_msgs/ByteMultiArray` message. However, sending out the packed bitstream proved to be much more efficient in terms of chirp efficiency η. The custom message type also allows to send out the configuration (RIC configuration string, number and length of FMCW sweeps) that was used to attain the

[2)] '['A' 10 'B']' is a quick way to print a newline character between 'A' and 'B'

message's echo. The message's timestamp and sequential ID is contained in the regular `std_msgs/Header`.

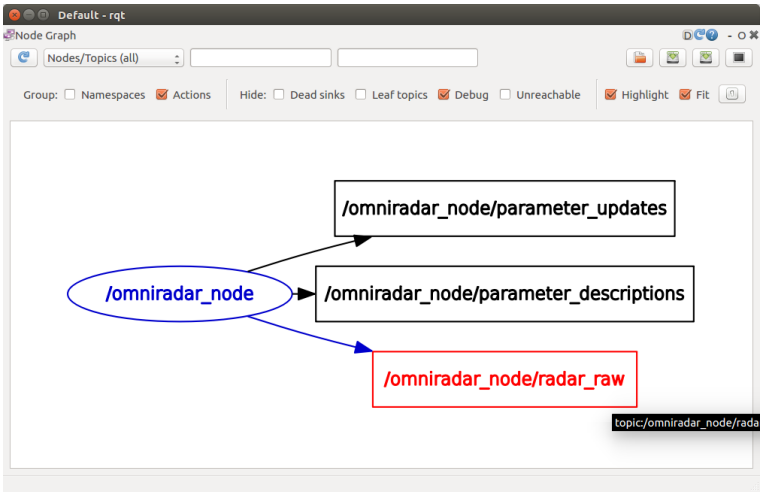


Figure 3.9: Omniradar rosnode node graph with leaf topics

While parameters from the ROS parameter server (as configured in the launch) are respected, the node also offers a dynamic reconfigure server to change RIC configuration string, number of sweeps and length of sweep on the fly without the need to restart the node.

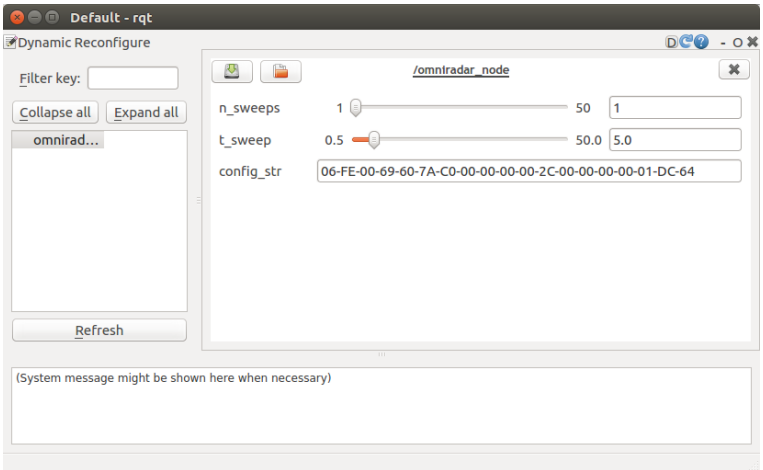


Figure 3.10: Dynamic reconfigure options for the Omniradar rosnode

The core of the node is a while loop that continuously triggers radar echo acquisition and copies the result into a new `omniradar::RadarEcho` message. The radar sensor update rate could be increased by offloading the message assembly and data copying into a C++11 lambda thread that is immediately detached.

```
std::lock_guard<std::mutex> lock_rdk(mtx_rdk);
auto t_echo = ros::Time::now();
```

```
auto p_echo = rdk->AcquireEcho(msg.n_sweeps);

std::thread t (
    [=] ()
    {
        std::lock_guard<std::mutex> lock_msg(mtx_msg);
        msg.header.stamp = t_echo;
        msg.packed_echo.resize(p_echo->size());
        std::copy(p_echo->begin(), p_echo->end(), msg.packed_echo.begin());
        pub.publish(msg);
        msg.header.seq++;
    }
);
t.detach();
```

The multithreading approach lets the node use between 100% and 140% CPU (observed with `htop`).

3.4 Matlab

Matlab (release R2017a) was chosen as implementation platform and language because it allows quick prototyping, provides relatively easy visualization, and, with the Robotics Toolbox, supports many ROS features.

3.4.1 Custom ROS message support

It is necessary to install custom message support with the roboticsAddon and generate the files to read in rosbags with `RadarEcho` type messages.

3.5 Complementing sensors

TODO

3.5.1 RGBD

3.5.2 Lidar

3.6 Implementation

3.6.1 Overview

3.6.2 Data path

3.6.3 Code structure

3.6.4 Usage

1. Start robot, connect three sessions with `ssh zero2-pa`
2. Start ROS core with `roscore`
3. Start the node, with keyboard teleoperation and Cartographer Laser Slam: `roslaunch omniradar omniradar_teleop_lidarslam.launch`
4. Use `rqt` with the *Dynamic Reconfigure* plugin to set the Omniradar node to generate the preferred number of sweeps and sweep duration. The configuration string can also be changed. The defaults are fine (One 5ms sweep)
5. `rosbag record /omniradar_node/radar_raw /odom /tf /map -O scan` will record a rosbag "scan.bag" containing all ROS messages with radar data, Kobuki odometry and slam map and transforms from Cartographer.
6. In the terminal running `roslaunch`, use the arrow keys to move the robot around (↑ and ↓ to increase and decrease speed, ← and → to increase and decrease rotation speed). *E* resets speed to zero and makes the robot halt.
7. After recording some interesting data, stop the rosbag record (*Ctrl*+*C*). Open a new terminal on your local machine and run `ssh zero2-pa "tar zcf - scan.bag" | tar zxf -`. The rosbag will be transferred to your machine. Using ssh with the tar pipe is the fastest way to transfer the data (around 100*Mbps* on the BSH wifi). Compressing first (`rosbag compress scan.bag`) and then sending takes a while on the not-so-powerful Odroid platform.
8. Optionally filter out unwanted transforms from the rosbag to speed up later processing: `rosbag filter scan.bag scan_filtered.bag '(topic == "/tf" and m.transforms[0].header.frame_id == "odom") or topic != "/tf"'`
9. In Matlab, run `radar_data = radar_bag2array("/path/to/your/scan.bag");`. This will read the bag sequentially into memory and extract the data: The robot position is recorded from odometry information and corrected using the /map → /odom transform as reported by slam localization. Cross range mileage is calculated as cumulative sum of distance between radar positions (as the radar is not mounted over the robot's rotation center, the radar mileage is different from robot mileage as

soon as any rotational velocity is present). Lastly, all values are interpolated at the radar message timestamps. It is a good idea to save the function's output, using `save('radar_data/radar_data_scan.mat', 'radar_data')`

10. The radar data can now be analyzed. The `plot_world_projection` script can be used to get a good overview over raw data, doppler speeds, direction of arrival, and reprojection map.
11. To compare radar reprojection map and laser slam map, try the `test_slam_overlay` with the correct bag filename (the lidar map is extracted from that bag)

3.7 Data Preprocessing

TODO

3.7.1 Rosbag to Matlab

3.7.2 (Odometry) Cross range interpolation

3.7.3 Raw Data Smoothing

After taking a single range reading, it will usually be relatively noisy. One solution to getting cleaner range data with higher SNR is oversampling. It is possible to use a moving average over a certain accumulation distance to achieve this. However, the number of raw samples is quite high and processing each sample takes a considerable amount of time (some minutes for some minutes of recorded data). It is better to make use of binning, with bins the width of the accumulation distance. All samples in one bin are averaged to represent that bin's value. This greatly improves processing time (to less than a second for some minutes of recorded data).

Figure #REF shows in gray the raw data of the first 20 range scan lines from the "Man-cave" dataset. It then compares different ways of averaging over these 20 scans. All of them increase the SNR, because a big part of the noise signal is statistically uncorrelated. Table #REF compares the RMS of the difference of each signal to the harmonic mean, which gives the cleanest signal.

Signal	RMS of difference to harmonic mean
Harmonic mean	0
Geometric mean	0.144
Arithmetic mean	0.275
50%-Trim mean	0.284
Median	0.318
Raw	1.064

The harmonic mean is defined as

$$harmmean(x_{1,2,\dots,n}) = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n}} = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$

The geometric mean is defined as

$$geomean(x_{1,2,\dots,n}) = \left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \sqrt[n]{x_1 x_2 \cdots x_n}.$$

The arithmetic mean is defined as

$$mean(x_{1,2,\dots,n}) = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

The 50% trimmed mean is defined as the arithmetic mean of all except the highest and lowest $\frac{n}{4}$ data points, where n is the number of data points. The median is defined as the value that lies between the lower and the upper half of sample values.

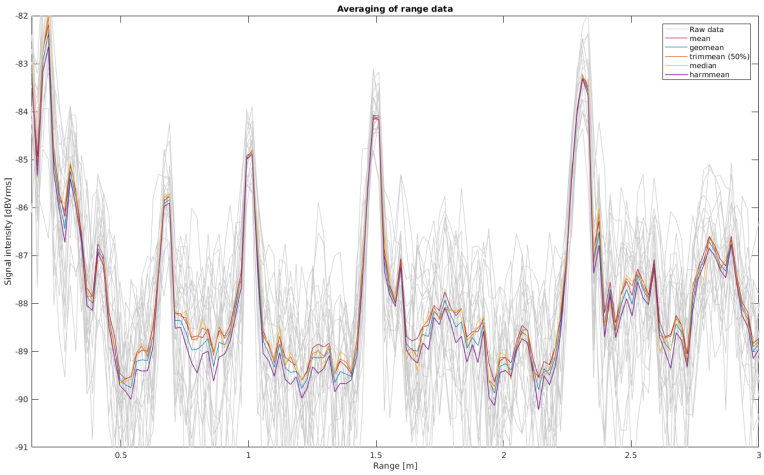


Figure 3.11: Range profile quality comparison with different averaging functions. Higher peak-to-trough ratio is better.

Due to the good signal quality, the implementation uses the harmonic mean to average the bins. Weighting the average with triangular or Gauss-shaped weight distribution did not noticeably improve data quality for any of the averaging methods.

Note that the range signal is not the only signal that needs to be averaged in a range bin. All other parameters that are part of the range scan need to be averaged as well. These parameters are mileage at scan time, robot position and orientation, and robot speed. Sweep time and down range bins don't change.

3.8 Doppler Estimation with the Peak Gradient Algorithm

The peak gradient algorithm is a way to find Doppler speeds from consecutive range profiles.

TODO put this in latex-subfigures?

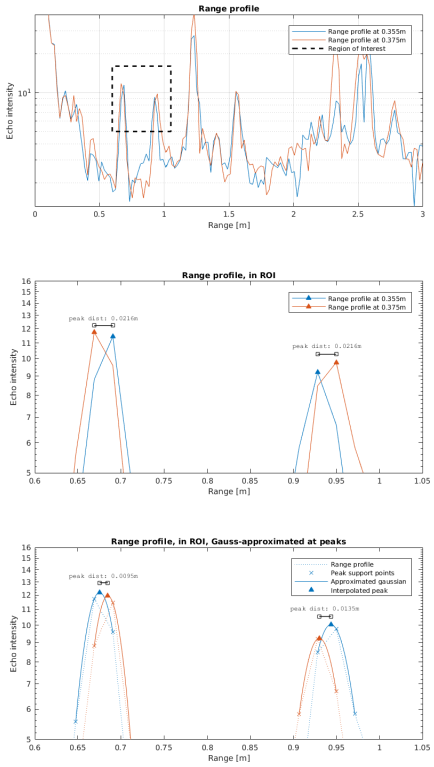


Figure 3.12: Peak detection and subsample peak interpolation to find peak distance between two range profiles

In figure ?? the scans at radar mileage $0.355m$ and $0.375m$ of a scene ("Basement") are overlaid. Obviously the range profiles of the two scans are very similar. However, as visible in figure #REF some peaks from target echoes are shifted, as the distance to the targets changes with the radar moving through the scene. The rate at which the distance to a target changes is its relative speed to the radar, the Doppler speed.

Usually, speed is measured in distance per time. In this case, it actually makes sense to ignore the scan's time stamps and look at the cross range (driven mileage) instead. With the Doppler speed as change of down range (distance to target) per cross range driven, calculations become time independent and hence radar movement speed independent.

A target's distance from the radar is assumed to be at the range bin of the corresponding peak in a scan's range profile. When a target's distance changes the peak will shift, too. This is visible in figure #REF and we can read the Doppler speed from the figure. The peak around $0.67m$ moves $d_{down,1} = 0.0216m$ closer in range, while the peak around $0.95m$ moves $d_{down,2} = -0.0216m$ closer (i.e., away). Combined with the change in cross range, $d_{cross} = 0.3752m - 0.3555m = 0.0197m$, we can calculate Doppler speeds of $v_{D,1} = \frac{d_{down,1}}{d_{cross}} = 109.55\%$ (of radar movement speed) and $v_{D,2} = \frac{d_{down,2}}{d_{cross}} = -109.55\%$. If the speeds were 100% and -100% it would mean that the targets are directly ahead and directly behind the moving radar. Speeds over 100% are however impossible in a

static environment where all relative target motion is caused by the radar movement. The targets are therefore either dynamic and moving by themselves, or the peak locations that determined those too-high speeds were not exact. Since we know that there were no dynamic moving objects in the controlled environment of the scan, the latter must be the case.

This effect of imprecise target peak localization and Doppler speed estimation could be overcome by averaging noisy data so that the average peak distance is close to the actual change in target range. A lot more scans are necessary for that though, and scan oversampling needs to be drastically reduced. This would lead to lower SNR, which means that some peaks with lower echo intensity could not be detected.

With higher downrange resolution, peaks could be localized more precisely. However, the down range resolution is limited by the available bandwidth of the radar sensor. In the Omniradar RIC60A, up to 7GHz are available, which is already extremely high. Its range resolution dR is $\frac{c}{2BW}$, which is roughly $2.1cm$. With this method, dR is of course the smallest measurable change of target range.

The localization of peaks is however not limited by range resolution, but by range accuracy, which mainly depends on SNR. It is much better than range resolution with $\sigma_R = \frac{dR}{\sqrt{SNR}}$. This can be utilized with subsample peak interpolation.

3.8.1 Inter-scan vs Intra-scan Doppler estimation

For correct Doppler estimation it is important to have the exact timing, or for relative Doppler speed, exact cross range mileage of the range scans whose peaks are compared in the peak gradient algorithm.

The Omniradar sensor can send multiple consecutive sweeps without any delay between them. The timing will then be very exact, because the precise length of one sweep is known. However, the number of sweeps in such a set of sweeps is limited (transmission of high data volumes will often fail), so they can't be average to be smoothed through oversampling and will be noisy. Smaller peaks will then not be detected reliably. Another problem with this approach is that it will give target Doppler speeds in change of down range over time, but not the robot speed-invariant relative Doppler speed in change of down range over change of cross range.

Inter-scan comparison gives better Doppler estimation, because the data can be smoothed through oversampling first. Consecutive sweeps can still be used: They need to be separated into individual scans with timestamps adjusted to $t_{msg} + i \cdot t_{sweep}$ (with message timestamp t_{msg}, consecutive sweep index i and sweep duration t_{sweep}) and cross range mileage interpolated at that timestamp.

3.8.2 Subsample peak interpolation

In subsample peak interpolation a curve is fitted on several supporting points in the coarse-resolution data. In the case of a single, non-overlapping radar echo peak, a Gaussian pulse of the form

$$g_i(x) = a_i e^{-b_i(x-c_i)^2}$$

is a good approximation. In figure #REF, the data point of the respective peak as well as its left and right neighbors are fitted with a Gaussian. The fit parameters a, b and c are calculated using Travis Wiens's crit_interp_g function.

As evident by visual inspection, the intensity and location of the fitted function's maximum are much closer to the real value.

With the same procedure as explained above, we measure peak distance shifts of $d_{down,1} = 9.510mm$ and $d_{down,2} = -13.52mm$ in figure #REF and can hence estimate Doppler speeds of $v_{D,1} = 48.26\%$ and $v_{D,2} = -68.63\%$ of radar movement speed. These values are a much more plausible estimation and generally work very well when used to calculate the reprojection angle in the reprojection method.

3.8.3 Peak matching

Visually it seems clear which peaks belong to each other in consecutive range scan lines. But for the algorithm this poses a challenge. Range scan lines usually don't have the same number of peaks because new targets can appear, old targets can disappear, noise can temporarily mask out targets peaks and target arcs cross each other. However if the cross range difference, i.e. the driven distance between scans, is not too high, a single target will not change very much in both range and echo intensity. This allows the detection of peak matches, between which the Doppler speed will be calculated. The parameters for peak matching tolerance are therefor allowed intensity change

$$max.ValueSearchArea = \delta \frac{I}{\Delta d_{cross}}$$

with intensity change factor δI and cross range difference Δd_{cross} allowed range change

$$max.LocSearchArea = \frac{\Delta d_{down} \Delta d_{cross}}{=} v_D = \alpha \; v_R$$

which is more easily expressed as maximum Doppler speed v_D in percent of robot speed v_R. Good values are a factor of $5cm^{-1}$ as maximum intensity change factor and a factor of $2v_R$ as maximum Doppler speed.

Minimum Peak height

3.8.4 Transmit crosstalk suppression

At low range, spurious peaks occur. The first one is caused by transmit antenna crosstalk and is visible as very high intensity echo around $d_{down} = 0$. After the transmit antenna crosstalk spike there is another peak around $d_{down} = 0.25m$ which is consistently visible. It can be explained with static objects sitting close to the radar, i.e. robot parts and the floor below the robot.

These spurious peaks create two problems: (1) automatic color scaling or height scaling respectively in plots is more difficult, and (2) high intensity false positives would be visible next to the robot path in the final map.

Hence these spurious peaks must be ignored during peak detection. This can be achieved in two ways. The first is to simply replace all data values under a down range limit d_{mute} (usually $d_{mute} = 0.30m$) with NaN values. For the second way, one effect of range compensation is exploited. As shown in figure #REF, the spike that previously had its maximum in the first range bin, at $d_{down} = 0$, now has its maximum in a later range bin. The reason is that the range compensation factor at $d_{down} = 0$ is $r(0) = 0$ but $r(d_{down} > 0) > 0$. As the transmit peak now has a maximum with neighbors lower than its maximum, `findpeaks` can find it. Color scaling can then be made to work correctly by clipping all values higher than the *second highest* peak. The Peak Gradient Algorithm also has an optional parameter `SkipFirstPeak` which, when set to *true*, ignores the first peak in each range scan line. This can help to ignore these echoes.

Note that the 0 value at $d_{down} = 0$ won't be displayed in log scale.

3.8.5 Peaks overlaps at crossing target arcs

3.8.6 Output

3.8.7 Limitations

Problems with imperfect fit function for subsampling

Problems with close targets

When Doppler speed is measured directly using FMCW, there will be several Doppler peaks, each representing a different target at the same range but with individual relative speeds. With the Peak Gradient Algorithm however, multiple targets at the same range are difficult to separate. In some cases this is only a temporary problem and is resolved by the radar moving a little farther so the ranges are separated by more than the range resolution. Sometimes however some peaks come from point-like targets that are close

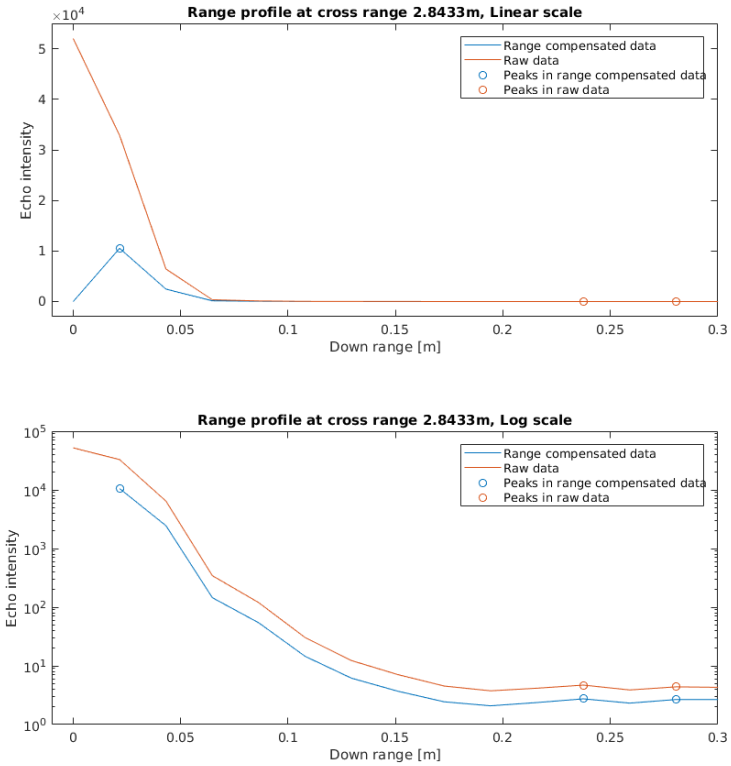


Figure 3.13: Attenuating effect of range compensation on transmit peak in range profile

together, like a parts of a wall. This bundle of targets is however not always separated by the same range. Especially in the case of a wall, the traces of visible points will cross each other as they slide on the sine arc (see figure). When the points are close together, only the brightest spots will be seen as peaks, and the trace of the detected peak matches will describe a squiggly motion. This causes the estimated Doppler speed to wander around the common speed. To combat this effect, a higher accumulation distance can be used during oversampling preprocessing, so the peaks move together so closely that they actually form a single target.

TODO subcaption with explanation of "simulation"

3.9 DOA Implementation

As described in #REF, the Direction of Arrival (DOA) angle can be measured from the phase difference at the receiving antennas of a multistatic radar.

In the case of RIC60A the antenna separation d is $1.16mm$ and wavelength λ is $\lambda = \frac{c}{60GHz} = 5.0mm$ (with speed of light c).

Figures #REF and #REF show the range profile and phase shift of the “Basement” scan. The phase shift is very noisy in the regions without a target peak in the range profile but

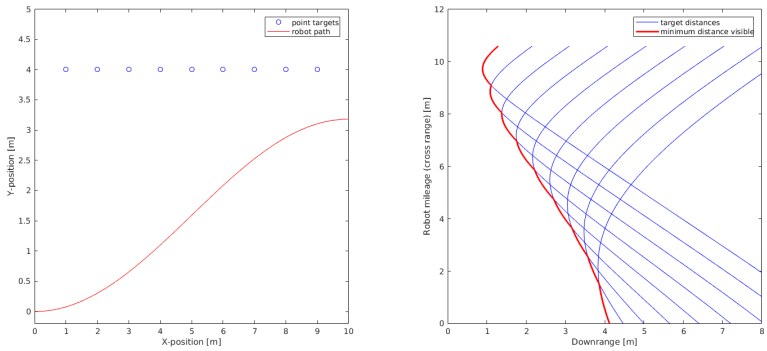


Figure 3.14: The target echo at the closest range usually has the brightest intensity. This can lead to errors in Doppler speed estimation.

exhibits steady values following a gradient over targets.

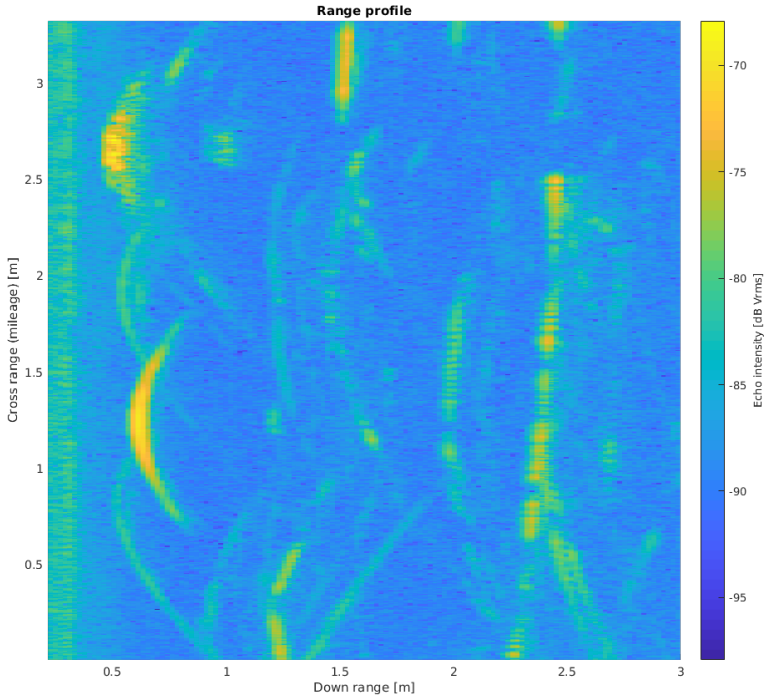


Figure 3.15: Range profile of Attic scan with color coded echo intensities at cross range over down range

Four steps are performed to get a reasonable estimation of direction of arrival.

Peak detection. In a first step, target peaks in the range profile are detected. The algorithm records for each peak in each range scan line its fitted interpolated location, full width at half maximum, and matching peaks in adjacent lines regarding value and location.

Down range averaging. In each range scan and at every detected target peak, the phase shift is averaged over the width of the respective detected peak. The average is weighted

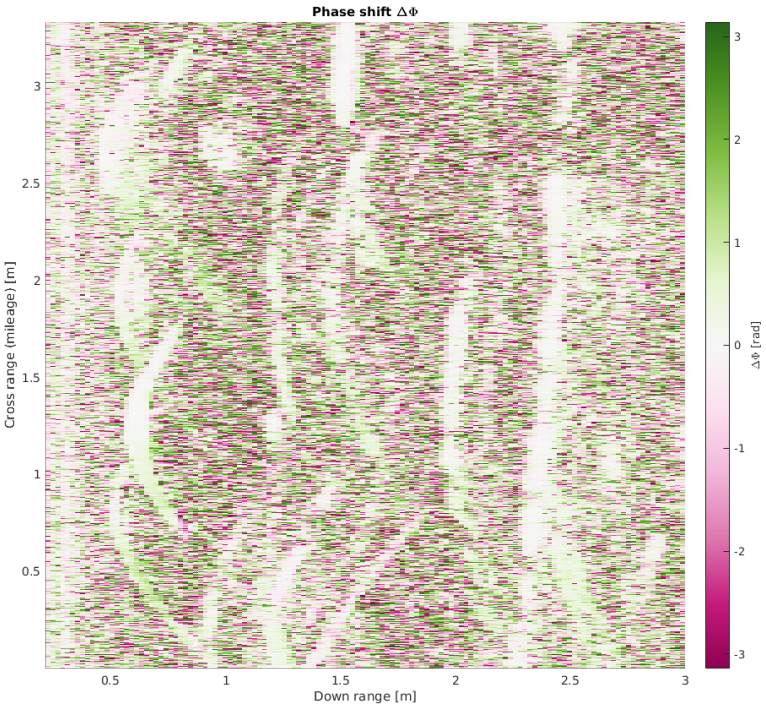


Figure 3.16: Phase profile of Attic scan with color coded inter-antenna phase difference at cross range over down range

using the Gaussian fit from subsample peak interpolation.

Cross range averaging. In every range scan line, each peaks phase shift is averaged over a configurable accumulation distance in cross range dimension. This is done by taking the arithmetic mean of all the phase shifts at all matching peaks (regarding value and down range location) within accumulation distance in cross range dimension.

Cut out. Noisy values at non-target peak range bins are masked out.

DOA calculation. In each scan line, each peaks direction of arrival θ is calculated from the smoothed phase shift values, using formula #REF.

Figure #REF shows the result of these four steps applied on the data of the "Basement" scan. The direction of arrival seems plausible: In this side-facing scan the robot passed some metal cans. Apparently the radar sensor was not mounted perfectly orthogonal to the robot's movement direction (which is not necessary for the reprojection method), but was slightly off by. This can be seen at the closest points of the target arcs. At the pericenter the line of sight to a target is orthogonal to the robot's movement direction, but the DOA value shows to be around 5 to 10 degrees.

Note that if the radar sensor is mounted inverted (rotated by 180°), DOA values have to be multiplied by -1 to keep right and left where they are.

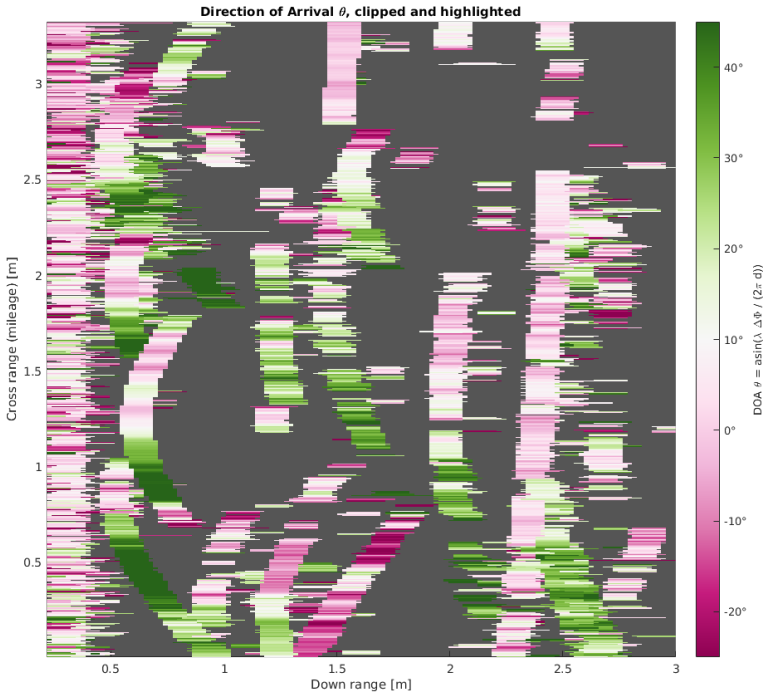


Figure 3.17: Direction of arrival estimation for Attic scan

3.10 Reprojection Mapping

3.10.1 Orientation parameters

Before the reprojection can be executed, the physical orientation of the radar sensor needs to be known to the algorithm. In the implementation, the two boolean parameters `forward_looking` and `mount_inverted` control the behaviour. If the radar's squint angle and angle sensitivity are such that the field of view reaches both sides of the robot path, the `forward_looking` parameter needs to be enabled. This enables the processing of DOA data to find the sign of every target's reprojection angle i.e. if it is to the left or right side of the robot's motion path. If the radar is mounted in an upside-down configuration the squint angle is not affected, but if the DOA values are processed they need to be mirrored (multiplication with -1) because the left and right antennas are switched. Otherwise, targets will be projected to the wrong side of the robot's path.

3.10.2 Projection direction

The radar reprojection can be executed as forward or backward mapping. The proof of concept implementation has an optional parameter `ProjectionMethod` to switch between forward and backward mapping.

Backward mapping

If backward mapping is enabled, the reprojection algorithm still operates range scan line based, but iterates over each pixel in the map. While this is computationally much more intensive, it allows to add negative information by reducing the map's value at pixels that are known to not contain a target because the range scan line does not feature a peak at that range.

```
foreach range_scan_line in range_scan_lines
  foreach pixel in map
    pixel_angle = robot.get_angle_to(pixel)
    distance = robot.get_distance_to(pixel)
    if distance < max_range && pixel_angle in field_of_view_range
      range_bin = range_scan_line.interpolate_at(distance)
      if range_bin.has_peak
        peak_angle = range_bin.peak.doppler.to_angle
        if peak_angle == pixel_angle
          map.at(pixel).add(range_bin.value)
      else
        map.at(pixel).reduce_value
```

Forward mapping

In forward mapping, the reprojection algorithm iterates over each range bin in each range scan line. Detected peaks are cut out and reprojected to a position on the map which is calculated from relative Doppler speed, range, and robot position. The projection target coordinates don't usually fall exactly on the map grid points. The implementation uses sample splitting to distribute a value over the nearest pixels in this case.

```
foreach range_scan_line in range_scan_lines
  foreach peak in range_scan_line
    foreach range_bin in peak
      target_coords = get_coords(robot.position, range_bin.range, peak.doppler)
      weights, neighborhood = split_sample(target_coords)
      map.at(neighborhood).add(weights, range_bin.value)
```

3.10.3 Sample splitting

To avoid aliasing when projecting a pixel in the forward projection direction, the sample is split over the four closest map pixels. The split is weighted with the distance of the target coordinates to the closest pixel centers.

If the target coordinate is $p_{target} = (x_t, y_t)$, then the horizontal and vertical distributions

ν_h and ν_v, respectively, are

$$\nu_h = \frac{x_t - \lfloor x_t \rfloor}{\lceil x_t \rceil - \lfloor x_t \rfloor} \nu_v = \frac{y_t - \lfloor y_t \rfloor}{\lceil y_t \rceil - \lfloor y_t \rfloor}$$

The pixel weights $p_{x,y}$ are then

$$p_{\lfloor x_t \rfloor, \lfloor y_t \rfloor} = \nu_v \nu_h p_{\lfloor x_t \rfloor, \lceil y_t \rceil} = \nu_v (1 - \nu_h) p_{\lceil x_t \rceil, \lfloor y_t \rfloor} = (1 - \nu_v) \nu_h p_{\lceil x_t \rceil, \lceil y_t \rceil} = (1 - \nu_v)(1 - \nu_h)$$

, such that

$$\sum_{x \in \{\lceil x_t \rceil, \lfloor x_t \rfloor\} y \in \{\lceil y_t \rceil, \lfloor y_t \rfloor\}} p_{x,y} = 1$$

In the special case where $y\_t = \lceil y\_t \rceil = \lfloor y\_t \rfloor$ there are only two instead of the four neighboring pixels. Their weights $p_{x,y}$ are

$$p_{\lfloor x_t \rfloor, y_t} = \nu_h p_{\lceil x_t \rceil, y_t} = (1 - \nu_h)$$

The same applies when $x\_t = \lceil x\_t \rceil = \lfloor x\_t \rfloor$:

$$p_{x_t, \lfloor y_t \rfloor} = \nu_v p_{x_t, \lceil y_t \rceil} = (1 - \nu_v)$$

And lastly, if $(y_t = \lceil y_t \rceil = \lfloor y_t \rfloor) \wedge (x_t = \lceil x_t \rceil = \lfloor x_t \rfloor)$, then $p_{x_t, y_t} = 1$

Figure 3.18: Illustration of a pixel value (white circle) being distributed over the four nearest grid neighbors. The shade of the neighbor's area indicates the individual distrution weight.

3.10.4 Range compensation

As evident from the radar equation #REF, echo intensity decreases with the fourth power of distance. This has the effect that reprojected targets appear brighter when they are mapped from close distance; but most importantly, when targets are detected from a far distance, mapped intensities are decreased due to the map's averaging.

This attenuation can be compensated with a range-based compensation factor f_r with

$$f_r(d_{down}) = \left(c_a + (\frac{d_{down} c_b}{+} c_c)^{-4} \right)^{-1}$$

Figure #REF shows the range profile of the "Torture Chamber" scan both as oversampled raw values (top subplot) and with range compensation enabled (bottom subplot). The middle subplot details the range scan line at one cross range highlighted by the red lines. Inspecting this detail graph reveals that both target peaks and noise floor are lowered in the near range, while the noise floor stays constant in the far range. This helps to keep a target's intensity at at least similar values over all ranges in the map.

TODO subfigures

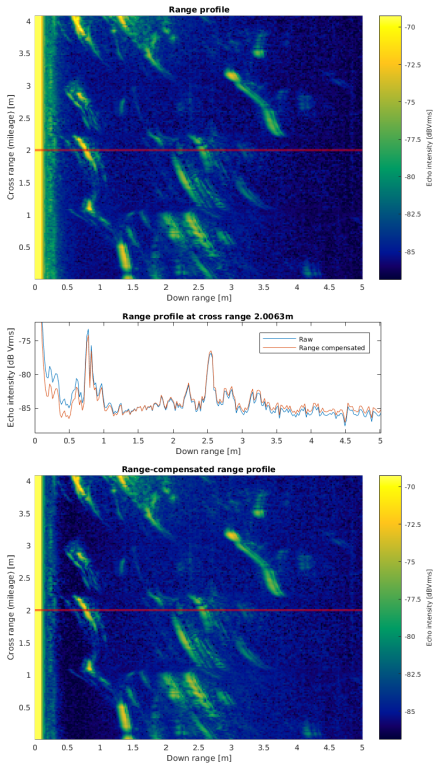


Figure 3.19: Range compensation

3.10.5 Angle sensitivity compensation

The intensity of a target peak depends on its angle with respect to the antenna. The angle is unknown before the Doppler speed is estimated, so the knowledge about echo attenuation caused by antenna angle sensitivity can not be used to improve peak detection. But the echo intensity influences how a target is represented on the reprojection map. Since the map averages all reprojections to any given point a low intensity echo will reduce the visibility of a target on the map. This can however be compensated by multiplying detected target peak heights with a factor that is based on the angle the target is believed to be seen under.

The angle compensation factor curve was found by experiment. A strong point target (a retroreflector) was placed at a known range away from the robot. The robot was then made to rotate around itself, such that the target comes into view and leaves again. Meanwhile, the radar scans, together with robot odometry were recorded.

The radar was not mounted over the center of rotation of the robot. This way, the radar did describe a circular path whose mileage can be calculated. The angle compensation measurement can hence be visualized in figure #REF in the usual range profile with echo intensities over cross versus down range. The range of the retroreflector varies with twice the distance of the radar to the robot's rotation center, but only the orientation is interesting - the range can just be summed up over the range bins the target is visible in

(in this case, $0.45m - 0.85m$).

In figure #REF, the same range scan lines are sorted by robot orientation during the scan. After the explained summing in down range dimension the intensity (absolute value of complex signal) data of both antennas is binned separately over 60 orientations from $-\pi$ to π.

The manufacturer later provided angle sensitivity measurements of the IC's batch (see figure #REF). The measurements show that the experimental approach produced valid results.

The compensation factor f_a for each angle is finally composed using the formula

$$m = \max\left(\max(s_{Rx1}), \max(s_{Rx2})\right)$$
$$f_a = \frac{1}{2}\left(\frac{m}{s_{Rx1}} + \frac{m}{s_{Rx2}}\right)$$

Multiplicating a peak which is to be reprojected at angle α with angle compensation factor $f_a(\alpha)$ results in a peak height that is independent of observation angle.

TODO subfigures

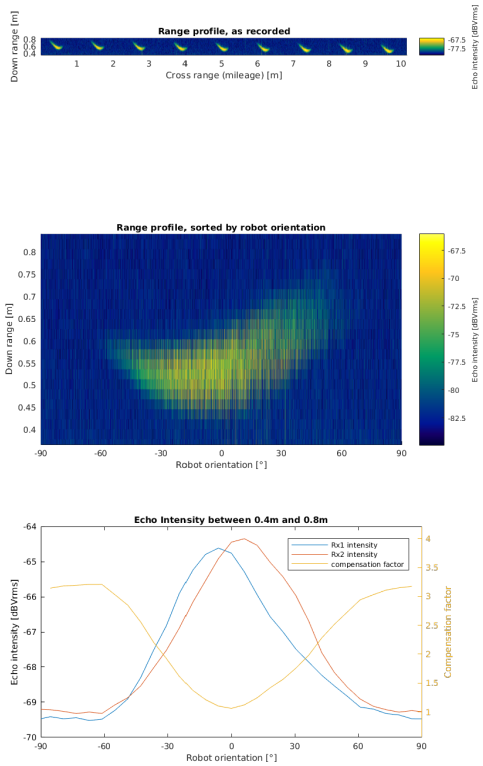


Figure 3.20: Angle compensation

3.10.6 Angle compensation window

Detected peaks are not in a single range bin, but form a curve over several range bins. Multiplicating each of these range bin's values with the same f_a does work, but leaves hard edges. It is better to multiplicate the peak with a window function with height f_a. The implementation the window w:

$$w(x, f_a) = 1 + (f_a - 1)\ e^{-\frac{(x-p_x)^2}{p_w}}$$

where p_x is the peaks subsample-interpolated peak location in down range space and

$$p_w = \frac{\left(\frac{fwhm}{4}\right)^2}{4ln(2)}$$

is the peaks width, where $fwhm$ is the full width at half maximum as found by the subsample-interpolated peak fit.

Figure #REF show a glass wall in the "Racetrack" scan. In the top subplot, only range compensation is applied. In the middle subplot, angle compensation is switched on. In the bottom subplot, both angle compensation and angle compensation windowing are switched on. The figure shows how angle compensation helps to keep the echo intensity of a mapped object at the same level, regardless of the angle at which it is seen by the radar.

TODO subfigures

3.10.7 World map resolution

TODO

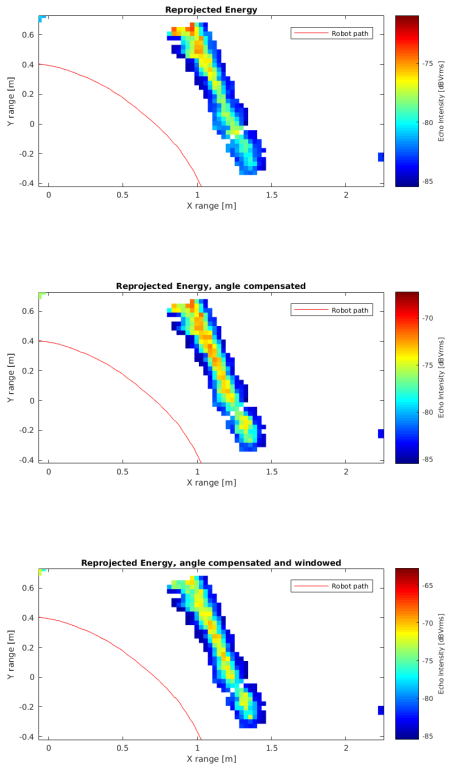


Figure 3.21: Effect of angle compensation and windowed angle compensation

4 Results and Evaluation

4.1 Evaluation

4.1.1 Evaluation dimensions

Needs to be * timely * false alarm rate * missed target rate * spatially precise * see different types of obstacles * comparable to other sensors * useful (humidity of wall is not interesting for obstacle detection)

4.1.2 Evaluation scan targets

- Cans (easy)
- Walls
- Glass walls
- Chair legs
- Cables on floor
- Cliffs

4.2 Results

During development of the reprojection method, over 30 scans were taken. The environment for the scans is the BSH office in the Bosch Research and Technology Center in Palo Alto. The office is fairly representative of a typical office environment. It has carpet floors, desks, office chairs, walls, corridors, and even glass walls.

The following list of scans is ordered by code name. It shows raw range scans (down range echo intensity vs. cross-range/mileage) for each scan, parameters of the scan (orientation, sweep time), and resulting map. For some of the scans, Lidar slam maps were recorded. They are overlaid in the background of the reprojection map.

...

5 Discussion and Comparison

5.1 Discussion

5.1.1 Reflection directionality

Many objects only become visible when their surface is oriented perpendicularly to the incident radar waves, so that enough scattered EM energy makes its way back to the sensor. This is very visible in the Underground scan, where a glass wall is detected as the robot passes it, but not while the robot sees it at an angle.

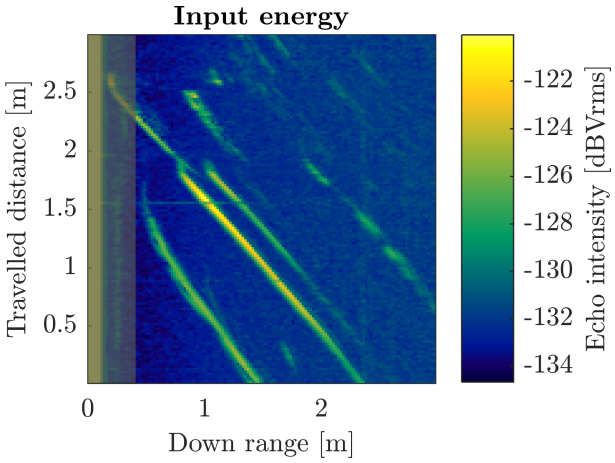
In the Torture Chamber scan, the same effect is visible for chair legs, especially for the chair at the scene center. the legs appear in clear form as soon as the radar sees the leg from a point that is orthogonal to the office chair legs.

5.1.2 Material-dependent echo strength

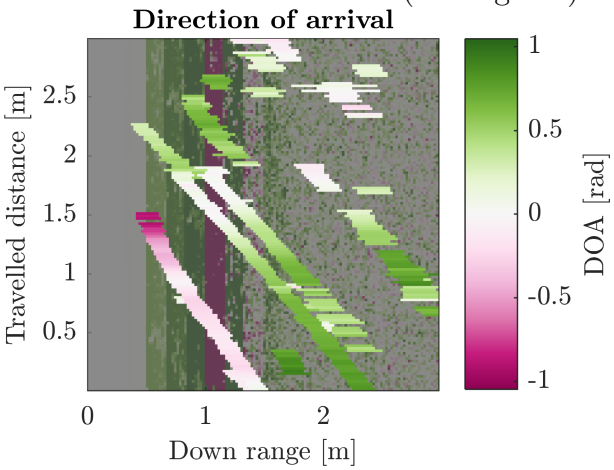
Some materials, like metal, are obviously better at reflecting radar waves than others, like styrofoam. Metal objects cause particularly strong echos which are visible from a higher distance. This can be observed in the hallway scans (e.g. Orbit, Public Restroom, Queue, Racetrack, Sauna, Underground), where the metal frames of doors and glass walls stand out in the scan.

5.1.3 Doppler vs Direction of Arrival data quality

In forward-facing geometry (scans D-T), the DOA is necessary to resolve the sign of a target's reprojection angle. This works fairly well, for example at the start of Sauna (see figure #REF), the closer target passes on the right (more pink), while the other targets stay



to the left side of the robot (more green).



In fact, for the side-facing case, the smoothed DOA data also turned out to be very good. It could even be used to calculate a more precise reprojection angle.

5.1.4 Multipath effects

Multipath effects are a well-known problem in ground-based radar applications [**Adams2012**]. In situations where multi-path effects are likely, there is a higher possibility that multiple versions of a target's echo are visible, which can lead to detection of incorrect angle and ranges. Luckily, in the recorded data almost no multipath effects are obvious. The only scan that shows some effects is the Torture Chamber. There, it seems like the radar waves bounce around a bit in the (2m,2m) area under the desk. The effect is that some targets are detected behind the wall behind the desk.

5.1.5 Object penetration

Some objects are penetrated by the radar waves. For example, in the Attic and Basement scans, both the front and the back wall of the plastic bottle can be seen. However, the plastic bottle was relatively close to the sensor. On the other hand, in the scans with

glass walls in them (P,P,Q,R,S,U), no significant radar echo is picked up from the (metal) chair legs behind the glass wall. This is because a typical glass pane attenuates the 60Ghz signal by about 5.5 dB [**Lu2014**]. In effect, a radar sensor with higher transmission power might be able to see through walls, but in the conducted experiments radar echos were to faint to be picked up after the first bigger object (like a wall).

5.1.6 Negative obstacles

With scans V,W,X the negative obstacle detection capability was analyzed. Cliffs, steps, and ditches are types of negative obstacles that cannot be traversed by the robot. In [**Jiang2015**], Jiang et al. claim that it is possible to detect this with UWB signals of various carrier frequencies. The experiments carried out for this thesis however did not show the same signal behaviour and it was not possible to detect cliffs.

The Virtual Reality scan was carried out in the standard configuration with a horizontal, slightly squinting, and not downwards angled sensor. The assumption was that a part of the strong signal int the 10cm to 20cm range were reflections from the floor, that should disappear when the floor ends at a cliff. Visual inspection of the range profile however shows only a extremely slight change in signal, e.g. at cross range 0.8m..1.1m, down range 0.2m..0.25m., where the floor could not reflect due to the radar sensor overhanging the cliff.

The Washroom scan has the sensor mounted in a vertical configuration and downward facing instead to increase sensitivity to echo scattering from below. The echo intensity for cross range 3.5m..4.5m is indeed just barely lower than 0m..2m, which maches up to where the sensor was over the edge and over floor, respectively.

TODO picture of scene

TODO annotation-arrow pointing to darker region?

To limit the transmit crosstalk's blinding effect, the sensor was mounted on a much higher position (on the RGBD camera mount) in the Xray Room scan. One effects is that the robot chassis itself is constantly visible at a distance of 0.35cm down range. At 0.45cm down range, the floor echo is visible. There is one dip in intensity at 2.5m cross range, where the sensor was not over floor. Overall the signal is however not as conclusive as in the Washroom scan.

TODO annotation-arrow pointing to darker region?

Maybe the signal could be improved with improved background subtraction (see #REF). However, the three scans show that it is very hard to detect negative obstacles with this sensor. A radar sensor of this type will hence not be a viable replacement for regular cliff detection sensors like the floor facing infrared distance sensors in the Kobuki base.

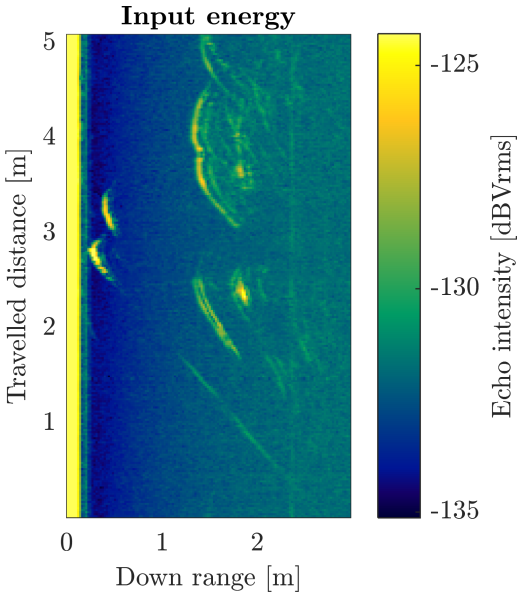


Figure 5.1: Input energy for the Washroom scan.

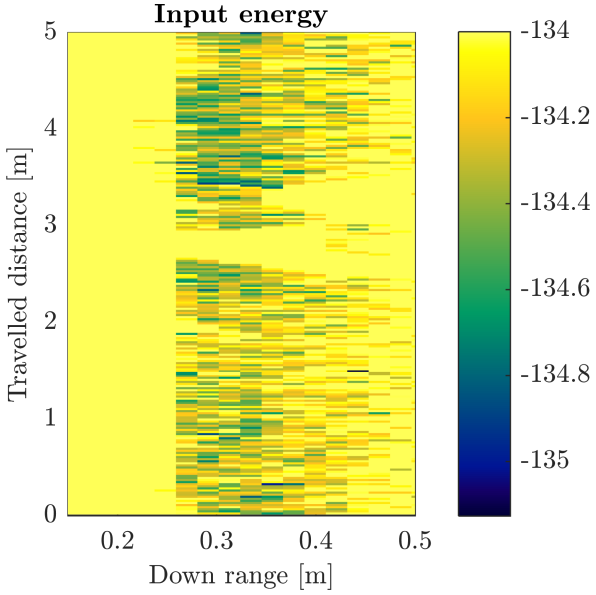


Figure 5.2: Input Energy of the Washroom scan, zoomed and intensity-clipped for better visibility of the effect.

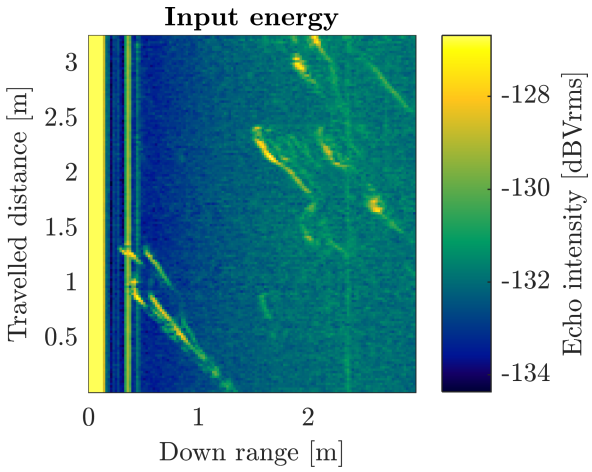


Figure 5.3: Input Energy of the XRay Room scan

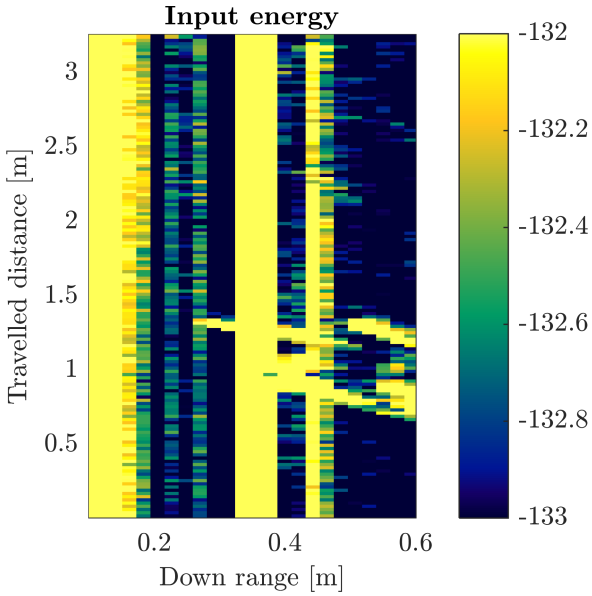


Figure 5.4: Input Energy of the XRay Room scan, zoomed and intensity-clipped for better visibility of the effect.

5.1.7 Cable detection

Cables on the floor are another interesting target that falls into the category of obstacles being a very common occurrence in the real world, but are hard to detect with conventional obstacle sensors. The Y (Is There A Cable On The Floor) scan deals with the detection if this kind of obstacle. For this, the same camera-mounted vertical configuration as in X Ray Room was used. Again, there is a constant robot chassis echo at down range 0.35. As the robot is driving closer towards the power cable on the floor, the cable's echo is visibly coming closer before it disappears under the robot's chassis. The echos at 0.9m down range show the two can towers at the end of the cable.

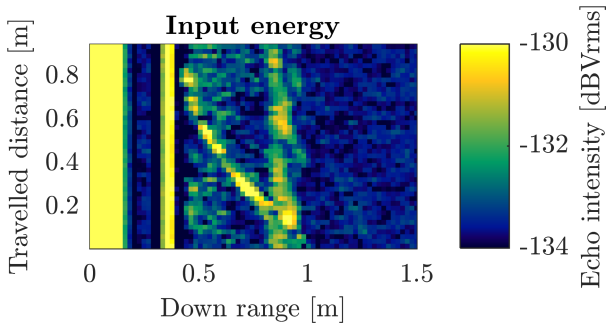


Figure 5.5: Input energy of the Y (Is There A Cable On The Ground) scan. The center arc represents the echo of a cable on the floor as the robot approaches.

Both the X Ray Room and Y (Is There A Cable On The Floor) introduce a new geometry that lifts the radar sensor out of the two dimensional mapping plane. The geometry is better described with a 3D case, for which more than

5.1.8 Minimum distance

The constant noise from the transmission crosstalk leads to a high minimum detection distance as explained in #REF. The effect is that targets can not be projected onto the map if the robot is too close to them. This is an issue in the Sauna scan, where the Glass wall right at the beginning of the robot path cannot be mapped in its entirety.

5.1.9 Parameter tuning?

5.2 Comparison with other mapping techniques

While some of the radar reprojection maps speak for themselves, they make more sense when compared to other mapping techniques. In the following, SAR techniques, Laser slam, and RGBD slam are compared to the radar reprojection.

5.2.1 SAR

Synthetic aperture radars make a lot of sense in other applications where the radar is moved over or through a map. The big difference to this application is that "professional" SAR applications have radar sensors that sit in vehicles that are not in the mapping plane. Airplanes, satellites and even Submarines scan the earth like that.

There are a few examples for UWB radars being moved sideways (usually on a rail) in an effort to scan a scene with synthetic aperture radar. Gregory L. Charvat's "tin can" radar [**Charvat2014**] might be the most famous one, with many examples at http://glcharvat.com/shortrange/. Another great resource was Henrik Forsten's Homemade Synthetic Aperture Radar, documented at http://hforsten.com/homemade-synthetic-aperture-radar.html. Forsten used the Omega-k algorithm [**Tolman2008**] and Stolt interpolation (#TODO: add book reference) to correct the range migration arcs.

TODO make quad-subfigure with Forsten's images?

Forsten was able to greatly improve his data quality by use of an minimum-entropy based auto-focusing algorithm. The trick with this is that the radar needs to move in a very straight line, where the "error in path linearity should be around less than tenth of a wavelength". In Forsten's radar, this is about 5mm. However with the 60GHz Omniradar this is around 0.5mm. Keeping a straight line with less than half a millimeter of linearity error is not realistically achievable on the Kobuki platform.

One big inherent problem with synthetic aperture radar algorithms is that basically all of them assume the radar to move in a straight line. While changing the squint angle helps to deal with issues such as earth curvature in satellite applications, SAR with curved or even arbitrary paths is a challenging topic, particularly because auto-focusing, which again relies on phase information, becomes more difficult [**Axelsson2002**].

5.2.2 RGBD

The Kobuki robot was also carrying a depth camera. Using the rtabmap Ros package, some 3D scans of the office environment where made.

5.2.3 Lidar

As stated in #REF, the Kobuki robot used in the experiments was equipped with an RPLidar and a computing platform powerful enough to perform slam. Lidar slam is the go-to, standard approach when it comes to mapping the environment around a robot. After years of research and product development, even cheaper lidar systems have acceptable range resolution. While they can't provide ground truth data (see problems with lidar data in #REF), it makes sense to compare the radar reprojection maps with laser scan maps.

6 Conclusion

6.1 Future Work

6.1.1 Dynamic target rejection

6.1.2 Online mapping

The proof of concept implementation processes pre-recorded data. However the algorithm is by no means limited to offline processing. Being very much iterative and range scan line based it requires only the knowledge of current and past, but not future scans. A live version of the algorithm was not built, because the implementation was done in Matlab, which does not run on arm processors natively. Matlab's Robotics Toolbox does include a way to receive ROS messages live, but it is very slow and would miss a lot of messages. This was tested by replaying a rosbag, which works only at less than 10% replay speed. This means that a 6 minute recording takes around 60 minutes to process. On the other hand, just reading in all the messages in a rosbag takes around 3 minutes, which is the reason the implementation was not done live. In a real system (i.e. not replayed from a rosbag) sending raw scan messages over the network requires a lot of bandwidth, so the sampling frequency also drops considerably.

An online system would probably have to be designed as a ROS node that runs on the embedded platform.

Another topic that needs to be looked at for an online version is the size of the reprojection map. It should automatically expand if a wider area is necessary. This could be handled in a nice way with ROS's `nav_msgs/OccupancyGrid` messages and/or the grid_map package. This would also allow pretty and useful visualizations with RViz.

6.1.3 ROS nodelet

The `omniradar_node` ROS node spends quite some time on copying the radar echo into the ros message that is to be sent out. As Austin Hendrix points out in ROS answers, "ROS doesn't provide intra-process, zero-copy publishing. Nodelets can be run multi-threaded, so it is possible to have zero-copy between different nodelets within a single nodelet manager". Recording a rosbag still involves copying the data, but in an online

system a nodelet-based reprojecting algorithm can be expected to bring a reasonable performance improvement.

6.1.4 Auto-thresholding

6.1.5 Realtime

An obstacle sensor's job is to provide information on impeding collisions before it's too late. Thus it would be great to have the system run under realtime constraints, so it can guarantee range scanning and reprojection mapping to be finished within a known time frame.

6.1.6 Dynamic changing of sweep time and bandwidth

6.1.7 Interference Investigation

Since the 60 GHz band is ISM, other sources can be present, e.g. the IEEE 802.11ad standard [**IEEE2014**] ("WiGig") which enables very high throughput wireless LAN operation in frequencies around 60GHz. Commercial products using WiGig are already available and could cause some interference with the 60GHz radar sensor. Detecting or avoiding interference in the range signal would be an important topic if this causes a lot of trouble.

6.1.8 Ultrasound

Usually, ultrasound sensors measure the distance to the closest object. However, K.C. Lee's project log of their "Sonar for the visually impaired" project [**Lee2015**] shows how cheap sensors can be hacked to read a range profile that looks very similar (see figure ??) to what is used for the radar reprojection method. It might be possible to adapt the algorithms in this thesis to use ultrasound sensors.

Most ultrasound range sensors use the time of flight of a pulsed echo, but FMCW-based ultrasound modules have been proposed [**Battaglini2014**] recently. With a range resolution of

$$dR = \frac{c_{sound,air}}{2BW} = \frac{330\frac{m}{s}}{2 \cdot 12.5kHz} = 1.32cm$$

the proposed sensor would be comparable to the Omniradar sensor's range resolution and accuracy. However, for measurements to be very accurate the speed of sound in air must be known as it depends on humidity and temperature [**Bohn1987**].

For forward-facing geometry DOA is necessary to resolve the sign of the reprojection angle. Sound waves do not not carry phase information, but with a transducer array, direction of arrival can still be estimated [**Kunin2010**].

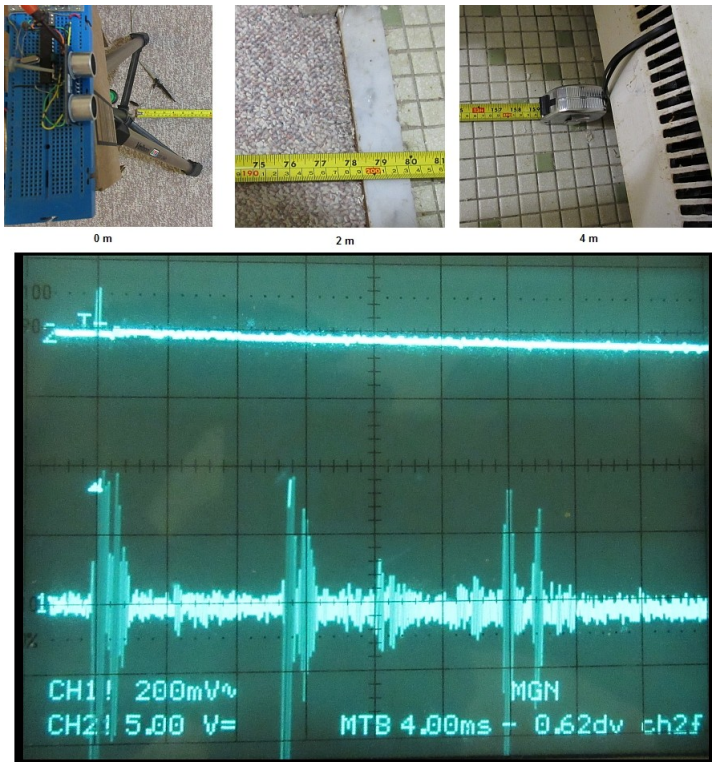


Figure 6.1: Range profile measurement with a hacked HCSR04 ultrasound module. Source: [**Lee2015**]

Extension to ultrasound sensors would be a very interesting topic for further work.

It might even be possible to use light waves with interferometric modulated flash lidars and the right optics.

6.1.9 3D case

Chapter 2.1 describes the geometry for 2D-cases. However, it is possible to extend the concept to the three dimensional space if a radar has more than two non-collinear receiving antennas, like visualized in figure **??**. Horizontal DOA would be measured between antennas at positions $Rx1$ and $Rx2$ while vertical DOA would be measured between $Rx1$ and $Rx3$.

Figure 6.2: Non-collinear triple receive antenna arrangement.

Reprojection mapping could then be used to build 3D occupancy maps. The reprojection angle α would then describe a circle at distance $R\,cos(\alpha)$ around the radar path, on which the detected target lies. Direction of arrival in horizontal and vertical planes would then have to be used to pinpoint the target location on the circle.

Figure 6.3: A target detected at range R and relative Doppler speed corresponding to reprojection angle α will be at a point on a circle (green) around the flight path (blue).

This holds also for the 2D-case: The possible locations for targets are then at the intersection of reprojection cone circle and floor plane.

This would be useful on vehicles moving in 3D space, like the TUM RCS's Modular Airborne Real-Time Testbed (MART) that was also used in other research and publications [**Becker2015**].

6.1.10 Single receive antennas on multiple sensors

Direction of arrival is a good solution to resolve the ambiguities presented in the general 2D-case 2.1.2 and 3D-case geometry (figure) for a sensor with a sufficient number of antennas. However, it should also be possible to use multiple, spatially separated sensors instead. Each sensor measures a reprojection angle. A target visible to both sensors can then be localized unambiguously with triangulation.

TODO

3D scenario. standard case: two intersecting spheres –> intersection circle doppler reprojection case: angle –> cones with intersecting base edge -> two interesection points (DOA to resolve, or high flying drone to assume)

6.2 Conclusion

Mobile robots still struggle to detect some obstacles that are invisible to their conventional sensors. Radar sensors have a big potential to improve obstacle avoidance. With reprojection mapping, this thesis proposes a novel method to create an obstacle map without having to steer the radar beam.

A proof-of-concept implementation of the reprojection mapping produced obstacle maps from the radar scans of the experiments for this thesis. The maps prove that some previously undetectable obstacles, like glass walls and office chair legs, can now be detected and mapped.

The next step on the way to a mobile indoor robot proficiently navigating real-world environments should be the implementation of an online version of reprojection mapping. This will also show if the results need to be further improved with more advanced noise rejection.

TODO

List of Figures

List of Tables