Seeing Around Street Corners: Non-Line-of-Sight Detection and Tracking In-the-Wild Using Doppler Radar

Supplemental Material

Nicolas Scheiner[*1] Florian Kraus[*1] Fangyin Wei[*2] Buu Phan[3]
Fahim Mannan[3] Nils Appenrodt[1] Werner Ritter[1] Jürgen Dickmann[1]
Klaus Dietmayer[4] Bernhard Sick[5] Felix Heide[2,3]

[1]Mercedes-Benz AG [2]Princeton University [3]Algolux [4]Ulm University [5]University of Kassel

1. Training and Validation Data Set

In this section, we describe the recorded data set and how it was generated.

1.1. Data Set Description

Our data set consists of 21 different scenarios including 2-8 repetitions each. In total the data set amounts to 100 sequences including over 32 million radar detection points. For our experiments we focus on two kinds of road users: pedestrians and cyclists. This choice is reasoned by the high effort for data recording and labeling. Also, the detection of bigger, faster, and more electrically conductive objects such as cars is in general much easier with radar systems. Therefore, we postulate that our results can be generalized and should even improve for such objects. For both kinds of recorded road users we measured and labeled roughly the same amount of sequences. We are mainly interested in the data from the two radar sensors installed in the front bumper of the car. Furthermore, we also record lidar data for reflector geometry estimation, camera images for scene documentation, as well as global navigation satellite system (GNSS) and inertial measurement unit (IMU) data for annotation purposes. The number of different scenarios was 21 for cyclists and 20 for pedestrians due to erroneous data in one scenario for all pedestrian repetitions. All measurements were recorded over five different days with five different people as rider and pedestrian, respectively. In each scenario we had the observed road user start at the ego-vehicle and drive alongside the reflector until out of range, cf. Fig. 1. Then, the cyclist or pedestrian turned and approached the test vehicle again on a similar trajectory. This allows us to effectively double our training data per scenario and, also, eases the quality checking procedure as the road user always start in the field of view of the documentation camera.

A complete data set overview is given in Tab. 1, where each scene is shortly described and the amount of repetitions for both object classes is indicated. Furthermore, in Fig. 3 some example pictures of used reflectors and road users are depicted.

In order to provide a better impression about the distances between the test vehicle, reflectors, and VRUs in the different scenarios, we plot their distribution in Fig. 4. We use the distance of labeled NLOS detection points for the estimation. While this approach yields an comprehensible way for determining the distances, it is affected by the irregular distribution of radar detections which is generally higher for objects in close distance. Thus, the shown graphs indicate rather a lower bound than absolute values of the actual distances. Moreover, a deeper insight in the data distribution is given in Fig. 5, where we compare the distributions of direct-sight and NLOS detection points on the VRUs in our data set averaged for every individual scenario. Despite the high number of NLOS detection points for the observed road users, the reduction compared to the direct sight object are obvious, ranging from a factor of roughly $2\times$ up to $50\times$ less detections. In comparison, the average total number of detections per measurement scene ($3.2 \cdot 10^5$) is three orders of magnitude higher than the combined number of direct- and non-direct sight detections, indicating the immense amount of clutter, the tracking algorithm has to cope with.

[*]Equal contribution.

1.2. Automated Labeling via Ground Truth System

Label acquisition is always a tedious and expensive task, in particular for abstract data representations. A human labeler is simply not as accustomed to radar measurements as to image data, for example. This problem is further increased when the objects of interest are non-line-of-sight objects. Fig. 6 gives an example view of one of the scenes recorded for this article. The radar data is displayed as bird's eye view (BEV) image before any custom filtering is applied. The difficulty of assigning the correct points to the VRU in all the clutter is easily recognizable from this example. It can further be seen from the corresponding camera image, that it is also very hard to utilize the optical labeling aid in the NLOS case, since the objects are often not present in the image data.

When allowing to limit parts of the data generation process to instructed scenarios the labeling process can be facilitated with the use of reference sensors. To this end, we equip vulnerable road users (VRU) with hand-held global navigation satellite system (GNSS) modules that are referenced to another GNSS module mounted on our vehicle (cf. Fig. 7). The positioning information can then be used to automatically assign radar data points falling within close proximity to the VRU's location. It can both be applied to direct radar measurements, but also non-line of sight measurement reconstructions. The reconstruction of non-line of sight measurements is explained in detail in Sec. 4. The final labels are subject to a manual quality assessment process where falsely assigned labels can be corrected in case of non-precise relay wall or GNSS information. The latter can often occur in scenarios with multiple high buildings around the measurement track. Due to the special task for this article to use buildings as reflectors, this step becomes more important than otherwise. Nevertheless, the utilization of the reference system allows us to obtain much more labeled data as would otherwise be possible with pure manual labeling. The system's setup and configuration are described in the remainder of this subsection. A similar system was described in [13]. The authors also empirically proved that the influence of the reference system itself on the radar measurements is negligible. In contrast to [13] we do not purely rely on GNSS data, but instead use the IMU for a complete pose estimation of the hidden vulnerable road users as described in [14].

Since we have both, GNSS and IMU available for our hand-held reference system, we opt for the method in [14]. Accordingly, the proposed reference system consists of the following components: VRU and vehicle are equipped with a device combining GNSS receiver and an IMU for orientation estimation each. VRUs comprise pedestrians and cyclists for this article. The communication between car and the VRU's receiver is handled via Wi-Fi. The GNSS receivers use GPS and GLONASS satellites and real-time kinematic (RTK) positioning to reach centimeter-level accuracy. RTK is a more accurate version of GNSS processing which uses an additional base station with known location in close distance to the desired position of the so-called rover. It is based on the assumption that most errors measured by the rover are essentially the same at the base station and can, therefore, be eliminated by using a correction signal that is sent from base station to rover. The VRU system comprises a GeneSys ADMA-Slim reference system installed in a backpack. Its IMU is based on micro-electro-mechanical systems (MEMS) gyroscopes and accelerometers, has an angular resolution of $0.005°$, and a velocity accuracy of $0.01\,\mathrm{m\,s^{-1}}$ RMS. The GNSS receiver has a horizontal position accuracy of up to $0.8\,\mathrm{cm}$. All system components except the antennas are installed in a backpack including a power supply enabling $\approx 10\,\mathrm{h}$ of operation. The GNSS antenna is mounted on a hat to ensure best possible satellite reception, the Wi-Fi antenna is attached to the backpack. Especially for the ego-vehicle, a complete pose estimation (position + orientation) is necessary for the correct annotation of global GNSS positions and radar measurements in sensor coordinates. For other applications such as tracking, the orientation of the VRU is also an

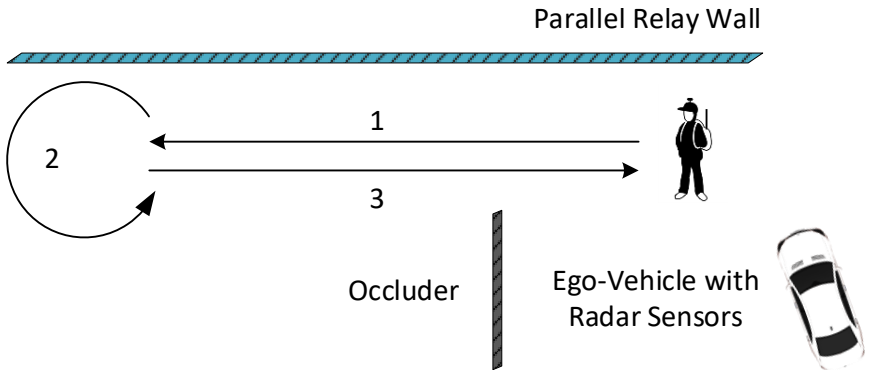


Figure 1: Scenario description: For all scenes, the road user moves alongside a reflective object (1), turns (2), and comes back on a similar path (3), effectively doubling the amount of training data.

Sc. #	Reflector Description	Cyclist Repetitions	Pedestrian Repetitions
1	Single parked car	2	4
2	Three cars parked in a row	3	4
3	Three cars parked in a row at a different location	2	4
4	Parked transporter van with trailer	4	2
5	Industry building with metal doors next to a garage	4	4
6	Corner of industry building without doors	2	1
7	Same industry building as above from a different position	2	2
8	Power distribution cabinet	2	2
9	Power distribution cabinet from a different position	2	0
10	Warehouse wall	2	2
11	Mobile standby office	2	1
12	Industry building - exit way	2	2
13	Multiple attached garages	2	2
14	Guard rail at intersection	2	3
15	Residential building	3	3
16	Plastered garden wall	2	2
17	Different residential building	2	2
18	Different three cars parked in a row	2	3
19	Different three cars parked in a row at a different location	3	2
20	Concrete curbstone at intersection	2	2
21	Small marble wall	3	3
	Total	**50**	**50**

Table 1: Data set overview: Short descriptions of all recorded scenes including the number of repetitions for each road user.

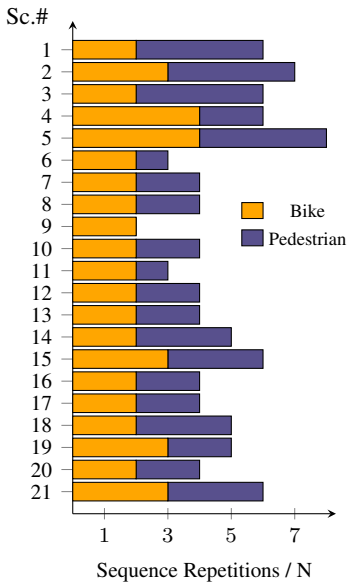


Figure 2: Data set sequence repetitions per scenario.

essential component. Furthermore, both vehicle and VRU can benefit from a position update via IMU if the GNSS signal is erroneous or simply lost for a short period. As vehicle reference system, we use a GeneSys ADMA G-Pro similar to the one in the backpack. Exact specifications for the GeneSys ADMA G-Pro can be found in the vehicle setup description in the Supplement Sec. 3.1.

Once all data from GNSS, IMU and radar is captured, VRU tracks have to be assigned to corresponding radar reflections. At first, all data needs to be transformed to a common coordinate system, e.g., car coordinates. Then, GNSS, IMU, and combined information is smoothed with a moving average filter of length 9 to remove jitter in positions and movements. The length corresponds to roughly $0.5\,\mathrm{s}$ for GNSS and $0.1\,\mathrm{s}$ for IMU data. Both durations are a trade-off between good smoothing characteristics and the expected interval of continuous VRU behavior. For each timestamp of each radar measurement the position is estimated by cubic spline interpolation. In the next step, an area around the position has to be defined. If ordinary car tracks were referenced with this method a rigid selection area could be easily defined as car orientation and outer dimensions can usually be estimated very precisely. For VRUs though, the selection area is more volatile. Swinging body parts or turning handlebars of cyclists, for example, complicate defining a fixed enclosing structure. Hence, two different versions of non-rigid surrounding shapes are proposed for the VRUs under consideration as depicted in Fig. 8 for pedestrians and Fig. 9 cyclists, respectively. Both shapes can be regarded as an approximation of the VRU's object signature, i.e., an accumulation of a multitude of measurements over time in a coordinate system with center and pose fixed to the VRU. The two object signatures depicted in Fig. 10 and Fig. 11 were obtained by measuring several repetitions of an eight-shaped trajectory to include all aspect angles. For both VRUs, those measurements were recorded using the GNSS+IMU reference system in a very clean environment. This way, all detections occurring in a generous area around the estimated VRU location can be freely added to its object signature without prior knowledge about its extends.

For a pedestrian, a bivariate Gaussian distribution is assumed according to Fig. 10. Thus an ellipse is used with its major axis oriented in movement direction (yaw angle) ϕ. Major and minor axis of the ellipse, $l_{\lambda,1}$ and $l_{\lambda,2}$, are calculated from fixed minimum pedestrian dimensions (empirically determined: $1.5\,\mathrm{m} \times 1.2\,\mathrm{m}$) plus a variable extra length for swinging

Figure 3: Example pictures from scenes in the data set. All images include either the observed pedestrian or cyclist. The scene ids corresponding to Tab. 1 are indicated in the captions. Please note, this figure allows for extensive zooming.

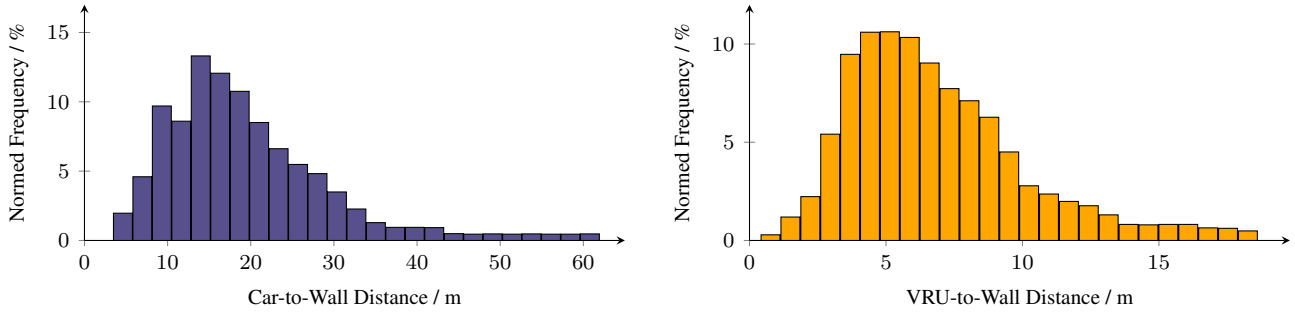


Figure 4: Histograms for distribution of distances between ego-vehicle and relay wall, and VRU and vehicle, respectively.

body parts defined by its velocity $\mathbf{v}$ and yaw rate $\dot{\phi}$:

$$l_{\lambda,1} = \begin{cases} 1.5\,\mathrm{m} + \min(\|\mathbf{v}\| \cdot 1\,\mathrm{s}, 1\,\mathrm{m}), & \text{if } \|\mathbf{v}\| \geq 0.05\,\mathrm{m\,s^{-1}}. \\ 1.5\,\mathrm{m}, & \text{otherwise.} \end{cases} \tag{1}$$

$$l_{\lambda,2} = \begin{cases} 1.2\,\mathrm{m} + \min(|\dot{\phi}| \cdot 5\,\mathrm{m\,s\,rad^{-1}}, 1\,\mathrm{m}), & \text{if } \|\mathbf{v}\| \geq 0.05\,\mathrm{m\,s^{-1}}. \\ 1.5\,\mathrm{m}, & \text{otherwise.} \end{cases} \tag{2}$$

Where $\dot{\phi}$ can be directly adopted from the IMU data.

The cyclist's object signature in Fig. 11 can be approximated with a rectangle. We use a rectangle with fixed length w_1 of

$2.5\,\mathrm{m}$ oriented in driving direction ϕ and width w_2 of $1.2\,\mathrm{m}$ plus a variable amount based on its yaw rate as labeling area:

$$w_1 = 2.5\,\mathrm{m} \tag{3}$$

$$w_2 = 1.2\,\mathrm{m} + \min(|\dot{\phi}| \cdot 5\,\mathrm{m\,s\,rad^{-1}}, 1\,\mathrm{m}) \tag{4}$$

As bikes usually cannot turn without driving, the derivation of ϕ assumes constant continuation of the cyclist's orientation for $\|\mathbf{v}\| < 0.05\,\mathrm{m\,s^{-1}}$ overruling the estimated yaw angle of the reference system.

At each time step, all radar detections that lie inside the defined regions are being assigned to the corresponding object instance label. Finally, all sequences are manually checked to ensure correct label assignments and make corrections where necessary.

2. Radar Image Formation

This section gives insight in the generation of the radar detection points which are utilized for object tracking. Therefore, we first derive how automotive radars resolve objects space, time, amplitude, and radial velocity according to [8]. Then, we show how the derivation is processed simultaneously on the whole measurement scene, and how object targets are detected in general. Lastly, we give an overview of how we combine the introduced techniques for our specific use case.

2.1. Radar Principle

Radar sensors send out electromagnetic (EM) waves which are reflected by close-by objects. Then again, the reflections are measured by the radar sensors. For objects to be "reflective", their physical micro structure size must be at least in the same order of magnitude as the EM wavelength λ. Furthermore, the reflector's conductivity is an important factor and increases its effectiveness. Modern automotive radar sensors typically operate in the frequency bands $76\,\mathrm{GHz}$ to $77\,\mathrm{GHz}$ and $77\,\mathrm{GHz}$ to $81\,\mathrm{GHz}$. The resulting wavelength can be directly calculated given the velocity of propagation which is the speed of light c:

$$\lambda = c/f. \tag{5}$$

The resulting wavelengths are between $3.70\,\mathrm{mm}$ and $3.95\,\mathrm{mm}$, thus, can well detect small structures. Nevertheless, tiny structures such as rain drops or fog particles still allow smooth propagation.

For this article, we utilize a frequency modulated continuous wave (FMCW) radar with chirp sequence modulation and a multiple input multiple output (MIMO) configuration. It can resolve targets in range r (i.e. distance), angle ϕ (i.e. azimuthal angle), and Doppler v_r (i.e. relative radial velocity). Though theoretically possible, we do not consider elevation angles here, since our sensor does not resolve targets by vertical differences. The measurement time is fixed for every individual sensor cycle which consists of N consecutive frequency chirps. Each frequency chirp consists of a continuous wave signal over a

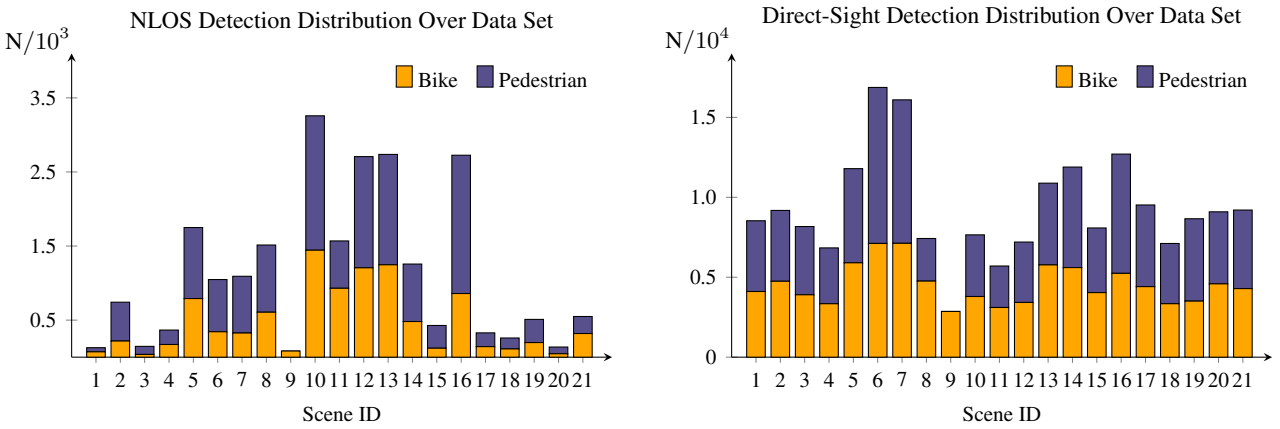
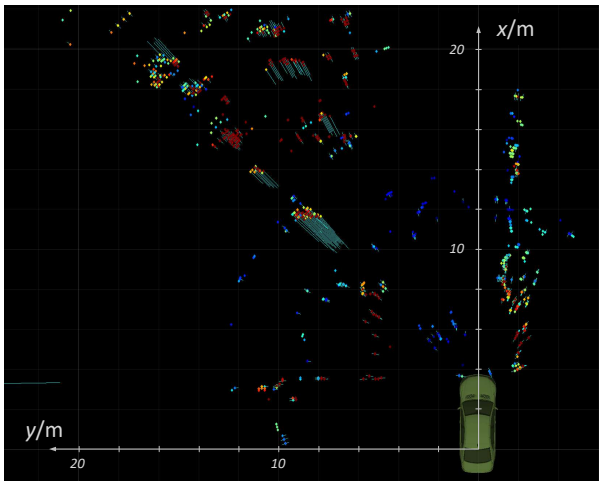
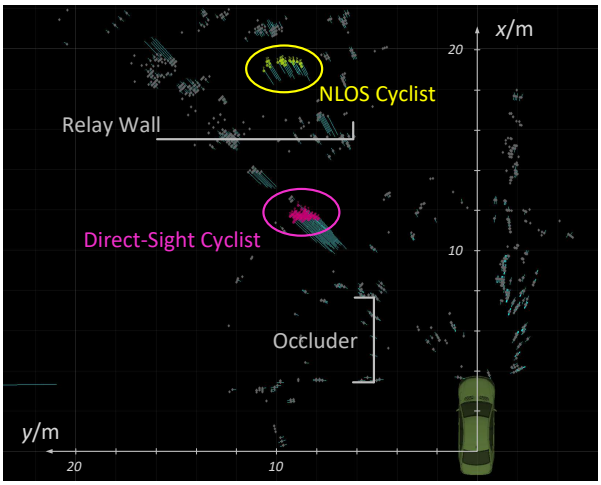


Figure 5: Data set statistics: distributions of labeled detections in the direct-sight and NLOS case, listed separately for both VRUs and averaged individually for each scenario.



Unlabeled BEV radar image. Amplitude values are color encoded, ego-motion compensated radial velocities are depicted by the cyan lines with their length representing the speed.



Labeled BEV radar image. Radar detections have been grouped and assigned with correct labels. The two objects of interest, the occluder and the relay wall, are highlighted in the image for better visibility.

Camera reference image of labeling scene at same time as radar images. VRU not visible in camera yet.

Camera reference image of labeling scene 400 ms later. VRU comes into sight of the camera.

Figure 6: Example pictures from labeling tool. The upper two images show the radar data in unlabeled and labeled view. The bottom row represents the images of documentation camera on the vehicle at the time of the radar data (left) and 400 ms later, i.e., when the object of interest comes into view.

frequency band B starting from the carrier frequency f_c, that is

$$g(t) = \cos\left(2\pi f_c t + \pi \frac{B}{T_m} t_c^2\right), \tag{6}$$

$$t = nT_{\text{tot}} + t_c \quad \text{with} \quad t_c \in (0, T_m), n \in (0, N-1), \tag{7}$$

with T_m and T_{tot} being the sweep's rise time and its total time. The instantaneous frequency of the signal in Eq. (6) is $1/2\pi \, d/dt_c \, 2\pi f_c t + \pi \, B/T_m t_c^2 = f_c + B/T_m t_c$, that is a linear sweep varying from f_c to $f_c + B$, which is plotted in Fig. 12. Beside the rise time and the total chirp, a pause time T_0 makes up for the difference according to Fig. 12. The emitted signal g propagates through the visible and occluded parts of the scene, that is this signal is convolved with the scene's impulse response. For a given emitter position $\mathbf{l}$ and receiver position $\mathbf{c}$, the received signal becomes

$$s(t, \mathbf{c}, \mathbf{l}, \mathbf{w}) = \int_\Lambda \alpha(\mathbf{x}) \, \rho(\mathbf{x} - \mathbf{w}, \mathbf{w} - \mathbf{x}) \, \frac{1}{(r_{\mathbf{lw}} + r_{\mathbf{xw}})^2} \, \frac{1}{(r_{\mathbf{xw}} + r_{\mathbf{wc}})^2} \, g\left(t - \frac{r_{\mathbf{lw}} + 2\, r_{\mathbf{xw}} + r_{\mathbf{wc}}}{c}\right) d\sigma(\mathbf{x}), \tag{8}$$

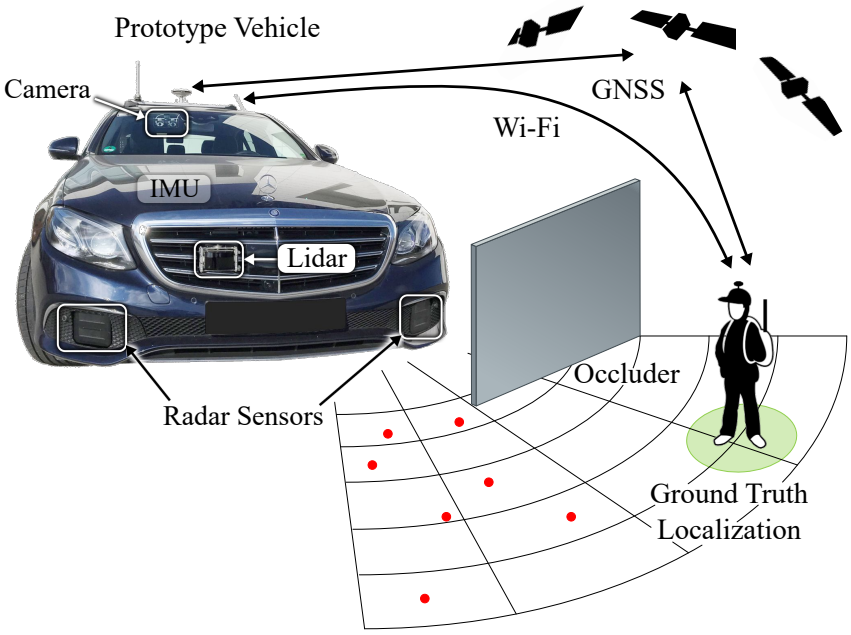


Figure 7: Automated label generation with a GNSS reference system: global positions of ego-vehicle and GNSS backpack are used to automatically assign labels to radar data in close proximity to the the backpack's location.

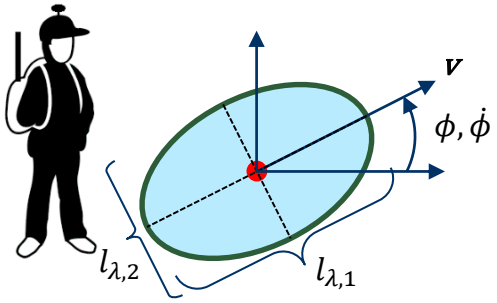


Figure 8: Pedestrian: Ellipsoidal selection area.

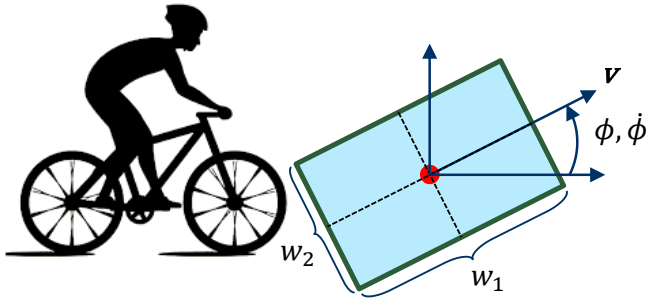


Figure 9: Cyclist: Rectangular selection area.

with $\mathbf{x}$ the position on the hidden and visible object surface Λ, the surface measure σ on the two-dimensional surface Λ, α as the ableto, and ρ denoting the bi-directional reflectance distribution function (BRDF), which depends on the incident direction $\omega_\mathbf{i} = \mathbf{l} - \mathbf{x}$ and outgoing direction $\omega_\mathbf{o} = \mathbf{x} - \mathbf{w}$. The distance function $r_{..}$ describes here the distance between the subscript position and its squared inverse in Eq. (8) models the intensity falloff due to backscatter.

The scattering behavior ρ depends on the surface properties. Surfaces that are flat, relative to the wavelength of $\approx$0.5 cm for typical 76 GHz-81 GHz automotive radars, will result in a specular response. We measure the surface roughness of the directly visible scene components using the first response of a scanning lidar system. As a result, the transport in Eq. (8) treats the relay wall as a mirror, see Fig. 13. We model the reflectance of the hidden and directly visible targets following [1], with a diffuse and retroreflective term as

$$\rho\left(\omega_\mathbf{i}, \omega_\mathbf{o}\right) = \alpha_d\, \rho_d\left(\omega_\mathbf{i}, \omega_\mathbf{o}\right) \; + \underbrace{\alpha_s\, \rho_s\left(\omega_\mathbf{i}, \omega_\mathbf{o}\right)}_{\approx 0}, \tag{9}$$

with $\omega_\mathbf{i}$ and $\omega_\mathbf{o}$ being the incident and output direction. In contrast to recent work [1, 5], we cannot rely on the specular component ρ_s, as for large standoff distances, the relay walls are too small to capture the specular reflection. Indeed, completely specular facet surfaces are used to hide targets as "stealth" technology [7]. As retroreflective radar surfaces are extremely rare in nature [9], the diffuse part ρ_d dominates ρ. Note that $\alpha(\mathbf{x})\rho\left(\mathbf{x} - \mathbf{w}, \mathbf{w} - \mathbf{x}\right)$ in Eq. (8) is known as the intrinsic radar albedo, describing backscatter properties, i.e. the radar cross section (RCS) [11].

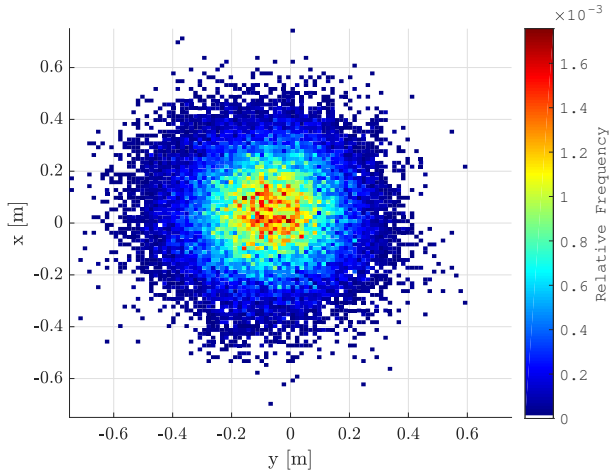


Figure 10: Pedestrian radar object signature calculated relative to GNSS+IMU position measurements.

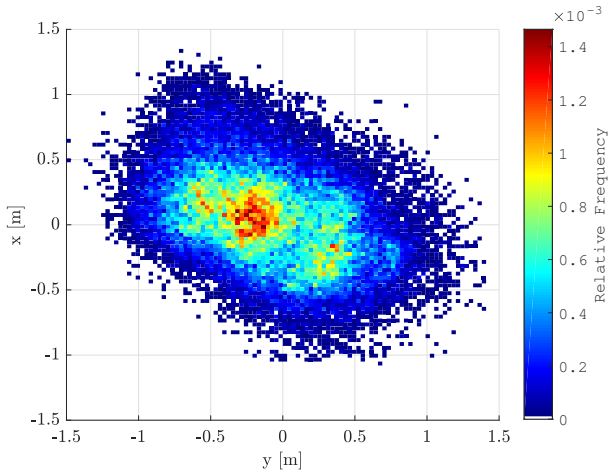


Figure 11: Cyclist radar object signature calculated relative to GNSS+IMU position measurements.

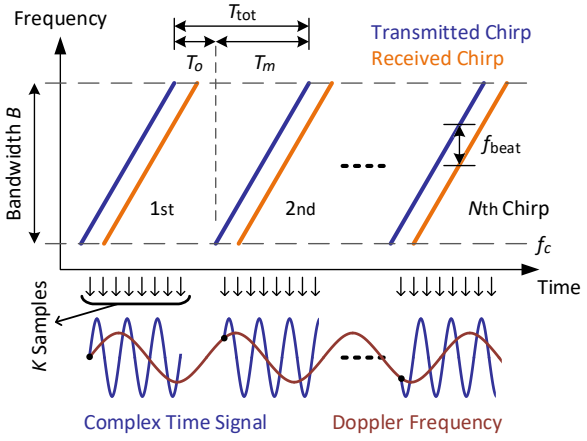


Figure 12: N chirp radar measurement.

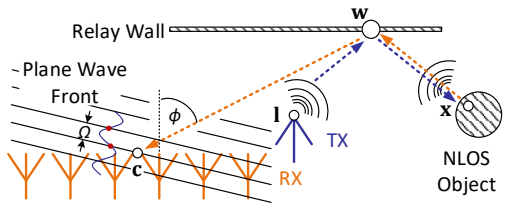


Figure 13: Uniform linear phased antenna array.

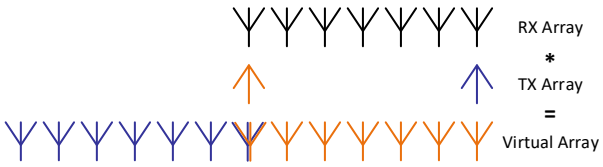


Figure 14: MIMO setup with 2 TX and 6 RX antennas.

Assuming an emitter and detector position $\mathbf{c} = \mathbf{l} = \mathbf{w}$ and a static single target ξ at distance $r = \|\mathbf{c} - \mathbf{x}\|$ with roundtime reflection

$$\tau_\xi = \frac{2(r + v_r t)}{c} = \frac{2(r + v_r(nT_m + t_c))}{c}, \tag{10}$$

Eq. (8) becomes a single sinusoid s_ξ:

$$s_\xi(t) = \alpha_\xi g(t - \tau_\xi), \tag{11}$$

$$\begin{aligned} p_\xi(t) &= s_\xi(t) \cdot g(t) \\ &= \frac{\alpha_\xi}{2} \cos\left(4\pi f_c t + 2\pi \frac{B}{T_m} t_c^2 - 2\pi f_c \tau_\xi - 2\pi \frac{B}{T_m} \tau_\xi t_c + \pi \frac{B}{T_m} \tau_\xi^2\right) \\ &\quad + \frac{\alpha_\xi}{2} \cos\left(2\pi \frac{B}{T_m} \tau_\xi t_c + 2\pi f_c \tau_\xi - \pi \frac{B}{T_m} {\tau_\xi}^2\right), \end{aligned} \tag{12}$$

where α_ξ describes here the accumulated attenuation along the reflected path. On a hardware level, the received signal s_ξ is mixed with the emitted signal g. Signal *mixing* corresponds to a signal multiplication, hence the resulting signal p_ξ consists

of the sum and a difference of frequencies, as $\cos(a) \cdot \cos(b) = 1/2\,(\cos(a+b) + \cos(a-b))$. The mixed signal is low-pass filtered, which removes the summation term in Eq. (12) from the signal, yielding

$$
\begin{aligned}
p_\xi(t) &\approx \frac{\alpha_\xi}{2} \cos\left(2\pi \frac{B}{T_m}\tau_\xi t_c + 2\pi f_c \tau_\xi - \pi \frac{B}{T_m}{\tau_\xi}^2\right) \\
&= \frac{\alpha_\xi}{2} \cos\left(2\pi \frac{B}{T_m}\frac{2r + 2v_r(nT_{\text{tot}} + t_c)}{c} t_c + 2\pi f_c \frac{2r + 2v_r(nT_{\text{tot}} + t_c)}{c} - \pi \frac{B}{T_m}\left(\frac{2r + 2v_r(nT_{\text{tot}} + t_c)}{c}\right)^2\right) \\
&= \frac{\alpha_\xi}{2} \cos\Bigg(2\pi t_c \Bigg(\underbrace{\frac{B}{T_m}\frac{2r}{c}}_{3.02e5-3.02e7} + \underbrace{\frac{2f_c v_r}{c}}_{5.07e1-5.07e3} + \underbrace{\frac{B}{T_m}\frac{2v_r t_c}{c}}_{6.67e0-6.67e2} - \underbrace{\frac{B}{T_m}\frac{4rv_r}{c^2}}_{2.01e-3-2.01e1} - \underbrace{\frac{B}{T_m}\frac{2v_r^2 t_c}{c^2}}_{2.22e-8-2.22e-4}\Bigg) \\
&\qquad + 2\pi n T_{\text{tot}} \Bigg(\underbrace{\frac{2f_c v_r}{c}}_{5.07e1-5.07e3} - \underbrace{\frac{B}{T_m}\frac{4rv_r}{c^2}}_{2.01e-3-2.01e1} - \underbrace{\frac{B}{T_m}\frac{2v_r^2 n T_{\text{tot}}}{c^2}}_{1.14e-5-1.14e-1}\Bigg) \\
&\qquad + 2\pi \Bigg(\underbrace{\frac{B}{T_m}\frac{2v_r n T_{\text{tot}} t_c}{c}}_{7.54e-2-7.54e0} + \underbrace{\frac{2f_c r}{c}}_{5.07e1-5.07e3} - \underbrace{\frac{B}{T_m}\frac{2r^2}{c^2}}_{1.01e-3-1.01e1} - \underbrace{\frac{B}{T_m}\frac{4v_r^2 n T_{\text{tot}} t_c}{c^2}}_{5.02e-10-5.02e-6}\Bigg)\Bigg) \\
&\qquad \text{with} \quad c = 3 \times 10^8\,\text{m}\,\text{s}^{-1},\ f_c = 76\,\text{GHz},\ B = 1\,\text{GHz},\ t_c = T_m \approx T_{\text{tot}} = 22.08\,\mu\text{s}, \\
&\qquad\quad 1\,\text{m} \le r \le 100\,\text{m},\ 1\,\text{m}\,\text{s}^{-1} \le v_r \le 100\,\text{m}\,\text{s}^{-1},\ n = N = 512 \\
&\approx \frac{\alpha_\xi}{2} \cos\left(2\pi\left(2\frac{B}{T_m}\frac{r}{c}t_c + 2\frac{f_c v_r}{c} n T_{\text{tot}} + 2\frac{f_c r}{c}\right)\right).
\end{aligned}
\tag{13}
$$

After plugging in Eq. (10) in the low-pass filtered signal and removing all components that would result in negligible values for real radar parameters, a compact approximation can be constructed. In the main paper we further neglect the non-constant phase term $2nT_{\text{tot}}{}^{f_c v_r}/_c$ as we only derive the range estimation there, leaving the Doppler components for the Supplement. From Eq. (13) it can be gathered, that the final signal approximation contains only three terms, a constant phase shift term, a frequency term, another phase term that is linearly changing over different chirps. The beat frequency of the frequency shift can be written as

$$f_{\text{beat}} = \frac{B}{T_m}\frac{2r}{c}. \tag{14}$$

The beat frequency is also depicted in in Fig. 12. It is much easier to measure than the time difference itself, thus, it is used for range estimation. The range can be estimated from this beat note accordingly, using

$$r = c\,\frac{f_{\text{beat}} T_m}{2B} \quad \text{with} \quad \Delta r = \frac{c}{2B}. \tag{15}$$

To this end, FMCW radar systems perform a Fourier analysis, where multiple targets with different path lengths (Eq. (8)) appear in different beat frequency bins.

Relative movement between the radar sensor and a reflective object results in a frequency shift in the received signal which is commonly known as Doppler shift:

$$f_{\text{Doppler}} = 2v_r/\lambda. \tag{16}$$

In order to avoid ambiguity between a frequency shift occurring from traveled distance opposed to relative movement, the ramp slope B/T_m is chosen to be very high, so that frequency shifts due to the Doppler effect are negligible in this case. Instead this information can be recovered by observing the linearly varying phase shift in the signals between different chirps of the sequence for each range bin separately. This corresponds to the second term of the final result in Eq. (13). The phase shift is also depicted in Fig. 12. More precisely, the Doppler frequency and velocity resolution can be estimated as function of the phase shift θ between two consecutive chirps:

$$v_r = \frac{\lambda \cdot \theta}{4\pi \cdot T_{\text{tot}}}, \quad \text{with} \quad \Delta v_r = \frac{\lambda}{2N \cdot T_{\text{tot}}}. \tag{17}$$

MIMO radars use several antennas for transmitting (channel input) and receiving (channel output) signals. Multiple antennas, for either receiving or transmitting signals, allow for using beamforming algorithms to estimate the angle of incidence. The most common form of antenna distributions are uniform linear arrays (ULA) which can resolve angles within all planes that include the array line, cf. Fig. 13. The angular resolution $\Delta\phi$ is given by the aperture size D which is defined as the distance between the two most distant antenna elements.

$$\Delta\phi \approx \lambda/D, \qquad \text{with} \qquad D = d\,(L-1), \tag{18}$$

where d is the antenna spacing, L is the total number of antenna locations, and λ is the signal's wavelength as defined by Eq. (5). Using conventional processing techniques, phased array antennas can separate a maximum of $L-1$ targets at any range and Doppler combination. Another estimate of the angular resolution can, thus, be gained by dividing the sensor's field of view (FoV) Φ by the number of separable objects

$$\Delta\phi \approx \Phi/(L-1). \tag{19}$$

Both definitions are, however, only accurate within the vicinity of the FoV center. At the edges of the FoV, the angular resolution deteriorates because of the reduced effective aperture size visible from these angle positions. For angle estimation a single TX antenna illuminates its whole FoV and all RX antennas receive the echo signal. Under a far field assumption, i.e., $r \gg \lambda$, the received signal arrives at the RX antennas as a plane wave as depicted in 13. An exact far field boundary is hard to describe, a common definition for the lower boundary is often set to

$$R_{\text{max}} = 2D^2/\lambda. \tag{20}$$

The signal phase shift Ω between two consecutive antenna elements can then be used to calculate the angle of incidence

$$\phi = \arcsin\frac{\Omega\lambda}{2\pi d}. \tag{21}$$

While theoretically only the two out-most antennas are important for angular resolution, in practical applications also the angular ambiguity range and, therefore, the spacing between elements is important. A common element spacing is $\lambda/2$, which is the largest spacing to fulfill the Nyquist-Shannon sampling theorem, i.e., to avoid ambiguities in the reception pattern. Smaller spacings would either reduce the aperture size D or increase the number of antenna element locations L, thus, increasing the hardware costs.

For MIMO processing multiple antennas for both transmitting (TX) and receiving (RX) elements are necessary. An example distribution is depicted in Fig. 14. Depending on which of the two depicted TX antennas is used, the relative distance between TX and RX antennas alters the reception pattern, effectively doubling the amount of receivers, but increasing the number of receive element locations to $\tilde{L} = 2L-1$ due to the overlapping middle elements. Mathematically this corresponds to a convolution of the TX antenna distribution with the pattern of the RX antennas

$$\mathbf{p}_{\text{virtual}} = \mathbf{p}_{\text{RX}} * \mathbf{p}_{\text{TX}}. \tag{22}$$

Where $\mathbf{p}_{\text{virtual}}$ are the virtual phase centers corresponding to the RX and TX antenna locations $\mathbf{p}_{\text{RX}}$ and $\mathbf{p}_{\text{TX}}$, respectively. Though, mathematically identical, the reduction of physical antennas results in a great reduction in costs and physical size. The combined array is commonly referred to as virtual array. For the receiver to be able to discriminate between TX antennas, their signals must be decoupled, e.g., by time division multiplexing. In practical applications one ore more overlapping elements in the virtual array are often used to correct phase and amplitude errors of the transmitted signals occurring during antenna switching. We use an antenna setup with a single overlapping element, which leads to an effective virtual array size of $\tilde{L} = 63$ element locations. More detailed information on MIMO radar processing can found, e.g., in [3].

Besides range, angle, and Doppler, all radar targets also have their own amplitude value. In automotive radar, noisy environments with high clutter amplitudes are common, therefore, absolute amplitude values are less important as for, e.g., Doppler. The received power at the RX antennas P_{RX} can formulated using the popular radar range equation

$$P_{\text{RX}} = \frac{P_{\text{TX}} \cdot G_{\text{RX}} \cdot G_{\text{TX}} \cdot \sigma \cdot \lambda^2}{(4\pi)^3 \cdot r^4}. \tag{23}$$

Where P_{TX} is the transmitted power, $G_{\text{RX/TX}}$ the antenna gain for RX and TX, and σ is the radar cross section (RCS), i.e., the equivalent reflection area corresponding to the illuminated object. This parameter has a high variance among different

objects, incidence angles, etc. A common procedure is to reformulate Eq. (23) and actually aim to estimate the RCS

$$\sigma = \frac{P_{\text{RX}} \cdot (4\pi)^3 \cdot r^4}{P_{\text{TX}} \cdot G_{\text{RX}} \cdot G_{\text{TX}} \cdot \lambda^2}. \tag{24}$$

The free-space path loss is then accounted for using r^4 correction. The degree to which this is accurate depend on how well the inherent assumption of isotropic reflection holds. Moreover, the RCS estimation in Eq. (24) does not account for violations in the free-space assumption.

The specifications of our prototype sensor are given in Tab. 2. The first row represents the emitted frequency range f and the operational bands for range (distance) r, azimuth angle ϕ, and radial (Doppler) velocity v_r respectively. The second row gives the resolutions $\Delta\cdot$ for time t, as well as for r, ϕ, and v_r.

Table 2: Radar sensor specifications as calculated from our sensor parameterization.

f/GHz	r/m	$\phi/\text{ deg}$	$v_r/\frac{\text{m}}{\text{s}}$
$76-77$	$0.15-153$	± 70	± 44.3
$\Delta t/\text{ms}$	$\Delta r/\text{m}$	$\Delta\phi/\text{ deg}$	$\Delta v_r/\frac{\text{m}}{\text{s}}$
100	0.15	1.8	0.087

2.2. Radar Intensity Falloff vs. Optical Falloff

Eq. (8) describes a major difference in the intensity falloff compared to existing optical approaches. For a point source at $\mathbf{c}$, the signal falloff is proportional to $1/(r_{\mathbf{lw}} + r_{\mathbf{xw}})^2 \cdot 1/(r_{\mathbf{xw}} + r_{\mathbf{wc}})^2$ instead of $1/r_{\mathbf{lw}}^2 \cdot 1/r_{\mathbf{wx}}^4 \cdot 1/r_{\mathbf{wc}}^2$ as the wall becomes specular for mm-waves. Note that low-cost automotive radars do not perform active beam steering but MIMO angle estimation, see previous subsection. Therefore, in contrast to optical NLOS methods, we do not focus RF power onto a single spot.

2.3. Radar Data Cube Processing Towards Ego-Motion Compensated Detection Points

Next, we describe the processing of the received time domain signal towards the point cloud which is the preferred data layer for most object detection and tracking algorithms. The signal processing is usually implemented on an FPGA and run internally by the sensor. However, in our case this pre-processing is done as a separate computation step in order to allow more flexibility in the processing order and parameterization.

Transition from Time Domain to Frequency Domain The radar data cube is a common way for the representation of received and sampled radar signal. For our sensor, it is a $K \times N \times L$ – more precisely $1024 \times 512 \times 64$ – dimensional data matrix, resembling the K samples per ramp, N ramps per sensor scan, and L antenna element locations. Given M object reflections of a radar scan, the received and down-converted signal is modeled as linear superposition of complex sinosoids:

$$\mathbf{y}(k,n,l) = \sum_{m=1}^{M} \alpha_m e^{j(\omega_{k,m}k, \omega_{l,m}l, \omega_{n,m}n)} + \eta(k,n,l). \tag{25}$$

The main part of this signal consists of a complex amplitude α_m which results from the reflection properties, i.e., the radar cross section. $\omega_{k,m}$, $\omega_{l,m}$, and $\omega_{n,m}$ are normalized radian frequencies corresponding to the three dimensions. $\eta(k,n,l)$ resembles a noise term. In the next step, the data cube is transformed to the frequency domain, where the three dimension directly represent range, angle, and radial velocity. As transformation operation often a discrete Fourier transformation (DFT) is utilized. The Fourier transformation can be implemented using a fast Fourier transformation (FFT) algorithm. The transformed signal after the i'th DFT can then be written as:

$$\mathbf{Y}^{(3)}(k,n,l) = \sum_{l=1}^{L-1} \sum_{n=1}^{N-1} \sum_{k=0}^{K-1} \mathbf{y}(k,n,l) \cdot h_K(k) \cdot h_N(n) \cdot h_L(l) \cdot e^{-j(\omega_k k, \omega_l l, \omega_n n)}. \tag{26}$$

ω_k, ω_l, and ω_n are now discrete frequencies in range, angle, and radial velocity.

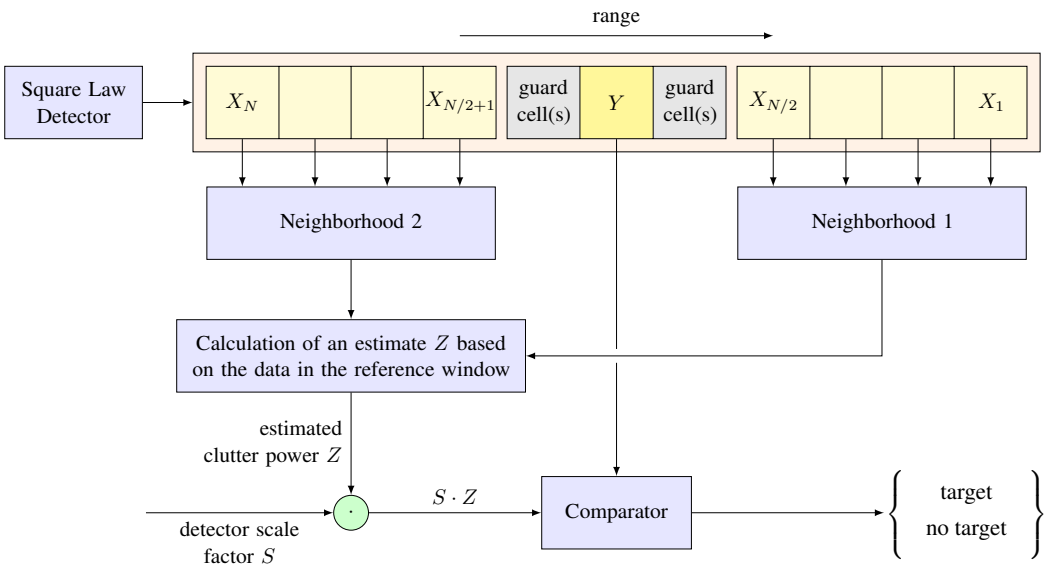


Figure 15: General architecture of range CFAR procedures. Figure copied from [10], Fig. 1.

Constant False Alarm Rate Filtering Once the radar data cube has been fully transformed to the frequency domain, i.e., is resolved in range, angle, and Doppler, standard procedure is to apply a constant false alarm rate (CFAR) filter in order to maximize the target detection probability in a noisy environment with a lot of clutter. To discriminate between noise, background reflections, and the objects of interest, aka. *targets*, the data is filtered using amplitude thresholds. As for real world applications, noise and clutter can vary heavily between measurements, adaptive thresholds become necessary. This adaptation to different environments is achieved by the CFAR detector. Its general architecture is depicted in Fig. 15 for the one-dimensional case. We illustrate the CFAR detector according to [10] and keep close to their nomenclature in order to match with Fig. 15, but highlight differences to previously used variables with subscripts. The basic principle is to use a fixed window size N_w in the direction of the examined dimension. The cell under test is in the middle of the window surrounded by two or more guard cells which serve to mitigate cell migrations effects, i.e., energy leaking to cells around the best-fitting one. The remainder of the window is used to calculate a clutter estimate $\mathbf{Z}$ which serves as basis for the final threshold level ϵ_{CFAR}.

Many different CFAR algorithms exist and a lot of them only differ in how they calculate the clutter estimate $\mathbf{Z}$. Two popular variants are the so-called cell averaging (CA) and the ordered statistics (OS) CFAR.

CA CFAR simply uses the arithmetic mean of amplitudes $\mathbf{X}$ (corresponds to $\mathbf{Y}$ in previous paragraph) among the N_w neighbors in cell the window.

$$\mathbf{Z}_{\text{CA}} = \frac{1}{N_w} \sum_{i=1}^{N_w} \mathbf{X}_i \tag{27}$$

OS CFAR, in comparison, first orders all N_w amplitudes according to their magnitude. The clutter estimate in this case is replaced by the k'th rank among amplitudes.

$$\mathbf{X}_{(1)} \leq ... \leq \mathbf{X}_{(k)} \leq ... \leq \mathbf{X}_{(N_w)} \tag{28}$$

$$\mathbf{Z}_{\text{OS}} = \mathbf{X}_{(k)} \tag{29}$$

In both cases, the final threshold level ϵ_{CFAR} is obtained by multiplications of the clutter estimate with an appropriate scaling factor S:

$$\epsilon_{\text{CFAR}} = S \cdot \mathbf{Z} \qquad \text{with} \qquad \mathbf{Z} \in \{\mathbf{Z}_{\text{CA}}, \mathbf{Z}_{\text{OS}}\}. \tag{30}$$

Finally, the threshold is used to obtain a sparse point cloud $\mathbf{U}$ from the data:

$$\mathbf{U} = \{\mathbf{x} \in \mathbf{X} \mid \mathbf{x} \geq \epsilon_{\text{CFAR}}\}. \tag{31}$$

The CFAR detector can be applied in this form in all three dimensions separately, or it maybe be combined to a 2D + 1D or even 3D algorithm. Multidimensional CFAR detectors naturally have a broader perspective across domains, however, they are much more computationally complex. For practical automotive applications it is important to note, that moving targets are generally easier to detect than stationary ones. In real-world automotive scenarios, there is always a lot of background reflectors, e.g., houses, trees, but at least the ground which maybe an even road in the best case or an uneven off-road scenario with many stones etc. This results in a lot of energy, i.e., high amplitudes, residing areas close to zero-velocity (ego-motion neglected) of the processed data cube. Hence, it is easier for a moving target to exceed the CFAR threshold level, thus, leading to a detection.

Ego-Motion Compensation One important advantage of radar sensors is their ability to directly measure relative movement, i.e. Doppler velocities, at very small hardware costs. For in-the-wild NLOS detections in road scenarios, sensing around corners must start while approaching a hidden object on collision course. In order to make full use of the radar sensors' capabilities, the measured radial velocities v_r have to be compensated for vehicle ego-motion during the measurement. Let v_{ego} be the signed absolute velocity and ϕ_{ego} the yaw rate of the ego-vehicle measured from the center of the rear axle of the car. For positive values, v_{ego} always points perpendicular to the rear axle towards the front of the car, i.e., only has an x-component in a car coordinate system. A more elaborate overview of coordinate system is given in Sec. 3. Then the velocity at the sensor $\mathbf{v}_{\text{sensor}}$ can be calculated as:

$$\mathbf{v}_{\text{sensor}} = \begin{pmatrix} v_{\text{ego}} + m_y \cdot \phi_{\text{ego}} \\ m_x \cdot \phi_{\text{ego}} \end{pmatrix}, \tag{32}$$

where m_x and m_y represent the mounting position of the sensor in x and y corresponding to the car coordinate system. Then the ego-motion compensated radial velocity $\tilde{v}_{r,i}$ of the i'th detection can be expressed as the difference between the measured value $v_{r,i}$ and the radial component of the sensor velocity towards the corresponding radar detection point:

$$\tilde{v}_{r,i} = v_{r,i} - \mathbf{v}_{\text{sensor}}^T \cdot \begin{pmatrix} \cos(\phi_i + \phi_{\text{ego}}) \\ \sin(\phi_i + \phi_{\text{ego}}) \end{pmatrix} \tag{33}$$

We use an inertial measurement unit (IMU) to estimate v_{ego} and ϕ_{ego}. Note, it is also possible to use the sensory data itself for ego-motion compensation. Therefore, the assumption is made, that most of the detections of one radar cycle correspond to non-moving objects. Then, these detections can be used as an alternative derivation of the subtrahend in Eq. (33).

Sensor Specific Processing Chain The radar sensors used for this task are experimental radar sensors that – opposed to most commercially available radar sensors – do not include an internal detection processor. This gives us some additional degrees of freedom for these processing steps. Specifically, we combine the above described processing steps in the following manner:

1. Compute the two-dimensional FFT $\mathbf{Y}^{(2)}$ in the range and Doppler domain in accordance with Eq. (26).
2. Apply a two-dimensional OS CFAR detector based on range and Doppler domain. The detector is empirically set to $N_w = 20$ using the 70'th percentile for k.
3. For all remaining range-Doppler bins, i.e., the ones detected by the CFAR filter:
   - Zero-pad the data along the angular dimension. This results in finer angular cell spacing, i.e., higher resolution as the zero-padded version $\tilde{L} > L$.
   - Compute the third and final DFT in angular direction, resulting in a variant of $\tilde{\mathbf{Y}}^{(3)}$ which is already sparse in the range and Doppler domain.
   - Apply a CA CFAR algorithm in angular domain resulting in the final point cloud $\mathbf{U}$ according to Eq. (31). The detector is empirically set to $N_w = 30$.
4. Perform ego-motion compensation (Eq. (33)), RCS estimation (Eq. (24)), and transform the sensor data to a unified coordinate system as discussed in Sec. 3.2, Eqs. (39, 40).

The main advantage of this processing chain, is that by allowing an improved spectrum estimation in the angular domain compared to standard 3D FFT processing, this approach leads to reduced antenna side lobe levels. These would otherwise lead to additional unwanted detections of strong reflective targets. Moreover, the reduced amount of computational complexity enables real-time processing on a standard GPU, e.g., $4\,\mathrm{GB}$ GeForce GTX 760. The major part of computational saving stems from the utilization of a 2D instead of a 3D CFAR algorithm in step 2. Furthermore, the third DFT and CFAR is only computed on the range-Doppler cells, which actually exceeded the CFAR threshold levels in step (2), which drastically reduces the computational effort for this step as well.

2.4. Detection Processing

Once the sensor has been processed, we obtain one ego-motion compensated point cloud of detections for each radar sensor. To merge the data streams from multiple sensors all data is transformed to a common coordinate system. Usually data is accumulated over short time frames, which are at least as large as to fit all sensors of the setup in once. To accumulate data over multiple time steps, a second coordinate transformation has to compensate the test vehicle's ego-motion during the observed time frame. The complete transformation procedure is derived in Sec. 3.2.

After transforming all data point to a common coordinate system, we utilize the fact, that we are mainly interest in the detection of *moving* road users, i.e., $\|\mathbf{v}\| > 0$. Therefore, we deploy a filtering algorithm, which removes undesired detection points. We examined two different approaches. Let $\mathbf{U}$ be a CFAR filtered and ego-motion compensated radar point cloud consisting of less than 10^4 points:

$$\mathbf{U} = \left\{ \left(\tilde{\phi}, \tilde{r}, \tilde{v_r}, \tilde{\sigma}\right)_i \mid 1 \leq i \leq R \right\} \quad \text{with} \quad R < 10^4, \tag{34}$$

where every point is described by its angular and range position, its ego-motion compensated radial velocity, and the RCS level estimated from the received power amplitude. Then our two detection filtering processes can described as follows:

1. Pure velocity-based filtering: For this approach, we retain all detections $\mathbf{u}_i$ exceeding a certain velocity threshold η:

$$\tilde{\mathbf{U}}_1(\eta) = \left\{ \mathbf{u}_i \in \mathbf{U} \mid \ |\tilde{v}_{r,i}| \geq \eta \right\}. \tag{35}$$

The advantage of this method is its simplicity and extremely efficient implementation. We empirically choose a threshold level of $\eta = 0.1\,\mathrm{m\,s^{-1}}$, which filters out over $95\,\%$ of detection values. The drawback of this approach is, that filtering may falsely remove some detections belonging to actual road users. This is due to the discrepancy of real velocity $\mathbf{v}$ and radial velocity v_r, but can also occur due to stationary object parts such as the pedestrians supporting leg or the part of a wheel that touches the road surface. Furthermore, the filter is very susceptible to small variations in η and may quickly lead to filter under- or overperformance, i.e., filtering out too few or too many points.

2. Velocity-density-based filtering: To incorporate also those close-by detection points, our second filtering approach evaluates not only the radial velocity of each detection, but also the velocities of close-by points. The first filter criterion is identical to the first approach, i.e., all points exceeding an absolute radial velocity of η_1 are retained. Moreover, if another point with sufficient absolute velocity is found in close distance, both points remain in the pointcloud. Thereby, a maximum distance region γ around each point is selected according its own radial velocity. Finally, points are also retained, if the have

a certain number of neighbors θ within close distance:

$$
\begin{aligned}
\tilde{\mathbf{U}}_2(\eta,\gamma,\theta) = \Big\{ \mathbf{u}_i \in \mathbf{U} \;\Big|\; & (|\tilde{v}_{r,i}| \geq \eta_1) \\
& \vee \Big((\eta_1 > |\tilde{v}_{r,i}| \geq \eta_2) \;\wedge\; \exists \mathbf{u}_j \neq \mathbf{u}_i : \Big((|\tilde{v}_{r,j}| \geq \eta_1) \;\wedge\; (\Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_2) \Big) \Big) \\
& \vee \Big((\eta_2 > |\tilde{v}_{r,i}| \geq \eta_3) \;\wedge\; \exists \mathbf{u}_j \neq \mathbf{u}_i : \Big((|\tilde{v}_{r,j}| \geq \eta_1) \;\wedge\; (\Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_3) \Big) \Big) \\
& \vee \Big((\eta_3 > |\tilde{v}_{r,i}| \geq \eta_4) \;\wedge\; \exists \mathbf{u}_j \neq \mathbf{u}_i : \Big((|\tilde{v}_{r,j}| \geq \eta_1) \;\wedge\; (\Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_4) \Big) \Big) \\
& \vee \Big((\eta_4 > |\tilde{v}_{r,i}| \geq \eta_5) \;\wedge\; \exists \mathbf{u}_j \neq \mathbf{u}_i : \Big((|\tilde{v}_{r,j}| \geq \eta_1) \;\wedge\; (\Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_5) \Big) \Big) \\
& \vee \Big((|\tilde{v}_{r,i}| \geq \eta_3) \wedge \big| \{ \mathbf{u}_j \in \mathbf{U} \mid \Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_2,\; \mathbf{u}_j \neq \mathbf{u}_i \} \big| \geq \theta_1 \Big) \\
& \vee \Big((|\tilde{v}_{r,i}| \geq \eta_4) \wedge \big| \{ \mathbf{u}_j \in \mathbf{U} \mid \Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_2,\; \mathbf{u}_j \neq \mathbf{u}_i \} \big| \geq \theta_2 \Big) \\
& \vee \Big((|\tilde{v}_{r,i}| \geq \eta_3) \wedge \big| \{ \mathbf{u}_j \in \mathbf{U} \mid \Gamma(\mathbf{u}_i,\mathbf{u}_j) \leq \gamma_3 \;\wedge\; \eta_1 > |\tilde{v}_{r,j}| \geq \eta_3,\; \mathbf{u}_j \neq \mathbf{u}_i \} \big| > \theta_3 \Big) \Big\} \\
\text{with} \quad & \Gamma(\mathbf{u}_i,\mathbf{u}_j) = \sqrt{\tilde{r}_i^2 + \tilde{r}_j^2 - 2\tilde{r}_i\tilde{r}_j \cos(\tilde{\phi}_i - \tilde{\phi}_j)}.
\end{aligned}
\tag{36}
$$

This filter is more complex and also requires a more fine tuning of the involved parameters. It can, however, also be implement in a very efficient way using a DBSCAN clustering algorithm with, e.g., a k-d tree backbone [12]. We use an empirically determined parameterization of $\eta_1 = 0.5$, $\eta_2 = 0.375$, $\eta_3 = 0.25$, $\eta_4 = 0.125$, $\eta_5 = 0$, $\gamma_2 = 0.25$, $\gamma_3 = 0.5$, $\gamma_4 = 1.0$, $\gamma_5 = 1.5$, $\theta_1 = 51$, $\theta_2 = 101$, $\theta_3 = 26$. This setting still achieves a point reduction of over $90\,\%$, but is better prepared to preserve static points with potential VRU correspondence. Another major advantage is the increased flexibility of the absolute minimum velocity threshold η_1 compared to η in the first approach, which is more robust to noisy measurements.

We conducted experiments for both filter settings, but opted for the the first approach for inference, because of its simplicity and time-efficiency for practical applications. For training and annotation, we rely on the second, more complex filter.

3. Setup and Prototype Details

In this section, we provide a detailed description of the setup components and calibration procedures.

3.1. Setup Details

All measurements for this article have been recorded using the test vehicle depicted in Fig. 16. The car is equipped with several sensors as indicated in Fig. 16. Further information regarding the mounting positions of the components involved in the recordings for this article can be found in Fig. 17. Moreover, the car contains several computers and interfaces concerned with the sensor control and recording. The most important part about the setup are two radar sensors in the front bumper of the car. The sensors are two identical experimental FMCW radar sensors operating with chirp sequence modulation. They are highly flexible in their ranges of parameter settings and can be adjusted according to requirements, e.g., short range vs. mid range measurement. For this article we decide on a mid range configuration with a wide field of view (FoV), i.e., ideal for urban situations or intersections. An overview of their exact configuration during our experiments can be found in Tab. 2. The only other sensor that is directly involved in our perception system is an experimental four layer lidar sensor. In the proposed system it serves for the recognition of the reflector's geometry. All other sensors mostly serve to improve the data quality. Therefore, global navigation satellite system (GNSS) unit and inertial measurement unit (IMU) are used for ego-motion compensation of measured Doppler velocities, or for a positional compensation when accumulating multiple measurements. Furthermore, the additional sensors facilitate the data labeling process, e.g., Wi-Fi, GNSS, and IMU are used for automated labeling as described in supplement Sec. 1.2. The camera helps for label correction or general scene understanding during offline processing. The GNSS system uses two satellite systems (GPS and GLONASS) and operates with real-time kinematic (RTK) positioning. As described in Sec. 1.2, RTK is a differential version of GNSS processing which uses an additional base station with known location in close distance to the position of the object of interest. The comparison of the signals of both GNSS receivers allows to correct many errors in the signal, effectively allowing to reach centimeter-accuracy. The addition of the IMU complements the GNSS location with an orientation estimate and also helps to correct the locations in case of

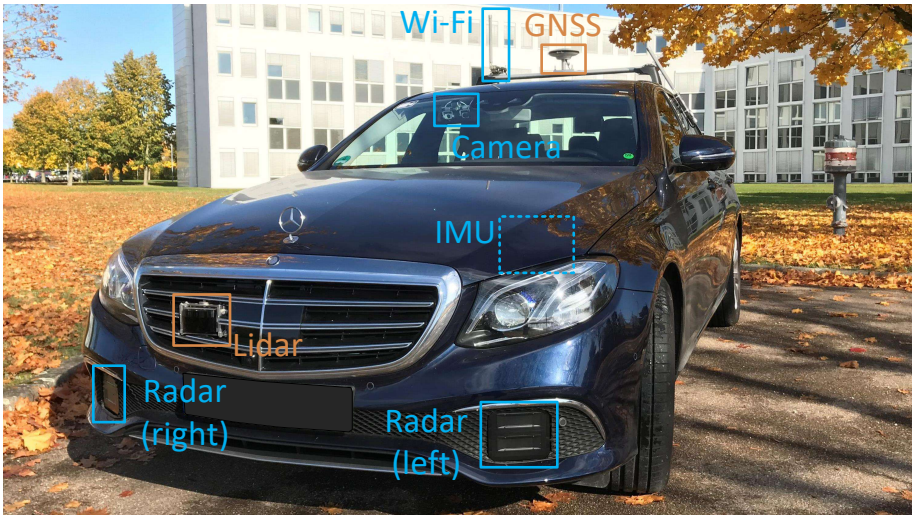
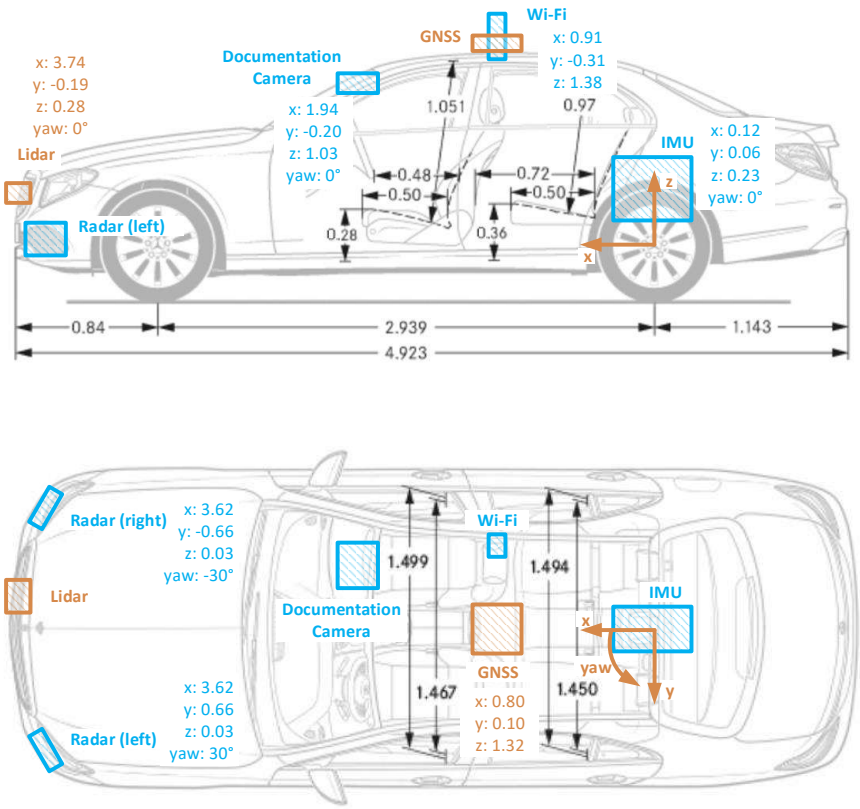


Figure 16: Prototype vehicle with measurement setup.

limited GNSS reception using an internal Kalman filter which fuses the signals from GNSS and IMU. All sensors including their key performance indicators can also be found in Tab. 3. We note, that a full sensor cycle over all sensors involved in the non-line-of-sight detection system, i.e., radar, lidar, IMU, and GNSS, takes only $100\,\mathrm{ms}$ and might even be accelerated to $50\,\mathrm{ms}$ using a different radar configuration. The low update rate of the camera is explicitly set to the same value in order to save storage.

Table 3: Specifications for all sensors on the ego-vehicle.

Sensor	Description
Radar	Experimental FMCW radar sensors with MIMO configuration and chirp sequence modulation: Frequency band: $76\,\mathrm{GHz}$ to $77\,\mathrm{GHz}$. Variable parameters depending on software configuration. The operational bands and resolutions of the chosen setting can a given in Tab. 2. Vertical FoV $10°$. Sensor cycle time: $22.6\,\mathrm{ms}$. Update rate: $10\,\mathrm{Hz}$
Lidar	Experimental 4 layer lidar sensor: FoV $145°$ horizontally and $3.2°$ vertically. Max range $327\,\mathrm{m}$. Resolution $0.25°$ horizontally and $0.04\,\mathrm{m}$ in range. Range accuracy: $<0.1\,\mathrm{m}$. Update rate: $25\,\mathrm{Hz}$.
IMU	GeneSys ADMA-G-PRO combined IMU and GNSS receiver: Three fiber-optic rotation rate sensors and three servo acceleration sensors. Angular resolution $0.005°$, velocity accuracy $0.01\,\mathrm{m\,s^{-1}}$ RMS. All values assuming settled Kalman filter. Update rate: $100\,\mathrm{Hz}$.
GNSS	GeneSys ADMA-G-PRO with integrated GNSS receiver: Novatel OEM-628. Reception of GPS and GLONASS. $4.5\,\mathrm{dBi}$ antenna gain, RTK processing with up to $0.8\,\mathrm{cm}$ horizontal position accuracy. Base update rate: $20\,\mathrm{Hz}$ with interpolation from Kalman filter fusing IMU and GNSS to $100\,\mathrm{Hz}$.
Camera	AXIS F1015 camera: Resolution 1920×1080 pixels, varifocal lens, wide angle mode with horizontal FoV up to $97°$. Update rate: $10\,\mathrm{Hz}$.
Wi-Fi	Omnidirectional tri-band antenna: Outdoor version, $5.5\,\mathrm{dBi}$ antenna gain, frequency bands at $2.4\,\mathrm{GHz}$ and $5\,\mathrm{GHz}$.



Adapted schematics from: https://www.mercedes-benz.de/passengercars/mercedes-benz-cars/models/e-class/

Figure 17: Car setup schematic.

3.2. Calibration Procedure

This subsection deal with error correction, initialization, and coordination of different sensors.

Radar Calibration Phase and attenuation inaccuracies by the radar sensors have been measured by the manufacturer in an anechoic room. Complex correction factors are applied for each measurement scan directly after recording.

IMU and GNSS Initialization For the combined IMU and GNSS module to work properly, it needs to be initialized prior to the first recording. This mainly serves to settle the internal Kalman filter fusing both, IMU and GNSS, signals. The initialization procedure consists of three steps, estimated times are indicated in brackets:

1. Standstill phase (10 s)
2. Straight acceleration phase (< 5 s)
3. Dynamic driving phase (> 120 s)

During standstill the IMU uses its accelerometers to estimate its orientation towards the gravitational center of the earth, i.e., the cars pitch and roll. Doors must remain closed and drivers must not move during this phase. During the next phase the vehicle is accelerated over a preset threshold velocity in order to fix its yaw angle estimation. The acceleration has to be executed on a line as straight as possible, and higher velocities are preferred. This is due to utilization of the GNSS velocity for yaw angle estimation which will be more accurate for higher velocities. Finally, during the dynamic driving phase, the internal Kalman filter is supposed to settle. In this step, the errors made by the laser gyroscopes and the servo accelerometers

are estimated and corrected. For fast convergence this phase should mainly consist of narrow eight-shaped circles or zigzag lines, as well as acceleration and deceleration maneuvers. This procedure is followed for both pose estimation systems of ego-vehicle and VRU, respectively.

Common coordinate system The original representation of each measured radar detection is given in the sensor coordinate (sc) system. The utilization of several spatially distributed sensors required that all sensory data is transformed to a common car coordinate (cc) system for further processing. The car coordinate system is located in the center of the rear axle of the car as depicted in Fig. 17. The utilized radar sensor estimates only horizontal variations, thus we focus on the two-dimensional transformation in x and y coordinates.

Given a single radar detection point $\mathbf{p}_{\text{sc}}$,

$$\mathbf{p}_{\text{sc}} = (x_{\text{sc}}, y_{\text{sc}})^T, \tag{37}$$

we first expand the position vector to yield:

$$\tilde{\mathbf{p}}_{\text{sc}} = \left(\mathbf{p}^x_{\text{sc}}, \mathbf{p}^y_{\text{sc}}, 1\right)^T, \tag{38}$$

the transformation is performed by multiplication with an appropriate transformation matrix $\mathbf{T}_{\text{s2c}}$:

$$\tilde{\mathbf{p}}_{\text{cc}} = \mathbf{T}_{\text{s2c}} \cdot \tilde{\mathbf{p}}_{\text{sc}} \qquad \text{with} \qquad \mathbf{T}_{\text{s2c}} = \begin{pmatrix} \cos m_\phi & -\sin m_\phi & m_x \\ \sin m_\phi & \cos m_\phi & m_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{39}$$

where $\tilde{\mathbf{p}}_{\text{cc}}$ is the extended version of the detection point in car coordinates, m_x, m_y, and m_ϕ are the mounting offsets of the sensor in x and y direction, and yaw angle ϕ, respectively.

Similarly, points can be accumulated over multiple time steps by compensating the ego-motion of the vehicle during the same time frame. This common practice in radar processing as it increases the number of detection points for a scene. We call the respective coordinate system *frame coordinate system* (fc). The transformation matrix $\mathbf{T}_{\text{c2f}}$ and the transformation itself are almost identical to Eq. (39):

$$\tilde{\mathbf{p}}_{\text{fc}} = \mathbf{T}_{\text{c2f}} \cdot \tilde{\mathbf{p}}_{\text{cc}} \qquad \text{with} \qquad \mathbf{T}_{\text{c2f}} = \begin{pmatrix} \cos s_\phi & -\sin s_\phi & s_x \\ \sin s_\phi & \cos s_\phi & s_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{40}$$

where $\tilde{\mathbf{p}}_{\text{fc}}$ is the extended version of the detection point in frame coordinates, i.e. the car coordinate system at the beginning of the examined frame. Then m_x, m_y, and m_ϕ correspond to the traveled distance and change in orientation during the beginning of the accumulation frame and the measurement of the detection, again in x and y direction, and yaw angle ϕ.

4. Reconstructing Third-Bounce Objects and Estimating Velocities

In this section we describe in detail how to reconstruct a real detection $\mathbf{x}$ from a third bounce or virtual detection $\mathbf{x}'$. Furthermore we explain how to estimate the real velocity of a detection by assuming it moves parallel to the relay wall.

4.1. Third-Bounce Geometry Estimation

This is a detailed overview on how to reconstruct a detection point $\mathbf{x}$ from a virtual detection $\mathbf{x}'$ given a relay wall $\mathbf{p} = \mathbf{p}_2 - \mathbf{p}_1$ represented by its two endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$. The endpoints are computed by the relay wall estimation described in Sec. 5.4. To decide if $\mathbf{x}'$ is a virtual detection we have two criteria. First, we need to decide whether the radar sensor $\mathbf{c}$ and the detection $\mathbf{x}'$ are on opposite sides of the relay wall. Second, we check whether the intersection $\mathbf{w}$ of $\mathbf{x}'$ with the relay wall lies between the endpoints $\mathbf{p}_1$ and $\mathbf{p}_2$. See Fig. 18 for an overview.

To check whether $\mathbf{c}$ and $\mathbf{x}'$ are on opposite sides of the relay wall we use the signed distance. For this we need the normal $\mathbf{n}_\mathbf{w}$ of the relay wall. We use the convention that $\mathbf{n}_\mathbf{w}$ must point away from $\mathbf{c}$, again use Fig. 18 as a reference. If the signed distance $\mathbf{n}_\mathbf{w} \cdot (\mathbf{x}' - \mathbf{p}_1) \geq 0$ then $\mathbf{c}$ and $\mathbf{x}'$ are on opposite sides of wall. Next we calculate the intersection $\mathbf{w}$ using basic linear algebra

$$\mathbf{w} = \mathbf{c} + \frac{(\mathbf{p}_1 - \mathbf{c}) \times \mathbf{p}}{(\mathbf{x}' - \mathbf{c}) \times \mathbf{p}} (\mathbf{x}' - \mathbf{c}). \tag{41}$$

Where $\times$ denotes the 2D cross product $\mathbf{a} \times \mathbf{b} = a_1 b_2 - a_2 b_1 = \mathbf{a}^\perp \cdot \mathbf{b}$. This is also known as the *perp dot product* since it is the normal dot product where $\mathbf{a}$ is replaced by the perpendicular vector $\mathbf{a}^\perp$ rotated 90 degrees anticlockwise.

We decide whether $\mathbf{w}$ lies between $\mathbf{p}_1$ and $\mathbf{p}_2$ by checking if $\|\mathbf{w}-\mathbf{p}_1\| \leq \|\mathbf{p}_2-\mathbf{p}_1\|$ and $\|\mathbf{w}-\mathbf{p}_2\| \leq \|\mathbf{p}_2-\mathbf{p}_1\|$. To summarize: A detection $\mathbf{x}'$ is a virtual detection if the following is true

$$\mathbf{n}_\mathbf{w} \cdot (\mathbf{x}'-\mathbf{p}_1) \geq 0 \wedge \|\mathbf{w}-\mathbf{p}_1\| \leq \|\mathbf{p}_2-\mathbf{p}_1\| \wedge \|\mathbf{w}-\mathbf{p}_2\| \leq \|\mathbf{p}_2-\mathbf{p}_1\|, \tag{42}$$

otherwise it is in direct line of sight.

To reconstruct the original detection $\mathbf{x}$ from the virtual detection $\mathbf{x}'$ we make the assumption that all reflections are perfectly specular. Reconstruction is done by mirroring the incident ray $\mathbf{w}-\mathbf{c}$ along the normal $\mathbf{n}_\mathbf{w}$

$$\frac{\mathbf{w}-\mathbf{x}}{\|\mathbf{w}-\mathbf{x}\|} = \frac{\mathbf{w}-\mathbf{c}}{\|\mathbf{w}-\mathbf{c}\|} - 2\left(\mathbf{n}_\mathbf{w} \cdot \frac{\mathbf{w}-\mathbf{c}}{\|\mathbf{w}-\mathbf{c}\|}\right)\mathbf{n}_\mathbf{w}, \tag{43}$$

with $\mathbf{w}-\mathbf{x}$ being the outgoing direction. Rearranging the terms and using $\|\mathbf{w}-\mathbf{x}\| = \|\mathbf{w}-\mathbf{x}'\|$ yields the reconstructed detection

$$\mathbf{x} = \frac{\left(\mathbf{w}-\mathbf{c}-2\left(\mathbf{n}_\mathbf{w}\cdot(\mathbf{w}-\mathbf{c})\right)\mathbf{n}_\mathbf{w}\right)\|\mathbf{w}-\mathbf{x}'\|}{\|\mathbf{w}-\mathbf{c}\|}. \tag{44}$$

4.2. Third-Bounce Velocity Estimation

After reconstructing $\mathbf{x}$ from $\mathbf{x}'$ we can estimate the underlying velocity vector $\mathbf{v}$ from the radial velocity v_r of $\mathbf{x}'$. This is done by assuming that the underlying object, which reflected the detection $\mathbf{x}$, is moving parallel to the relay wall.

We want to recall that the radial velocity v_r of an object at point $\mathbf{x}$ relative to the sensor $\mathbf{c}$ is given by the dot product between the normalized vector $\mathbf{x}-\mathbf{c}$ and the velocity $\mathbf{v}$. However in our case we did not measure the radial velocity of $\mathbf{x}$ directly, but the radial velocity of the virtual detection $\mathbf{x}'$. Therefore we have to consider the vector $\mathbf{x}-\mathbf{c}$ instead of $\mathbf{x}'-\mathbf{c}$. Overall this yields a relation between the radial velocity v_r and the angle φ between $\mathbf{x}'-\mathbf{c}$ and $\mathbf{v}$

$$v_r = \frac{(\mathbf{x}'-\mathbf{c})\cdot\mathbf{v}}{\|\mathbf{x}'-\mathbf{c}\|} = \frac{\|\mathbf{x}'-\mathbf{c}\|\,\|\mathbf{v}\|}{\|\mathbf{x}'-\mathbf{c}\|}\cdot\cos(\varphi) = \|\mathbf{v}\|\cdot\cos(\varphi). \tag{45}$$

Note that φ depends on the directions of $\mathbf{v}$ and $\mathbf{x}'-\mathbf{c}$. Changing either direction of $\mathbf{v}$ or $\mathbf{x}'-\mathbf{c}$ results in an angle $\tilde{\varphi} = \pi - \varphi$ with $\cos(\tilde{\varphi}) = -\cos(\varphi)$. Using this insight we can ignore the directions of $\mathbf{v}$ and $\mathbf{x}-\mathbf{c}$ to calculate $\|\mathbf{v}\|$ by using the following formula

$$\|\mathbf{v}\| = \frac{v_r}{\cos\varphi} = \frac{|v_r|}{|\cos\varphi|}. \tag{46}$$

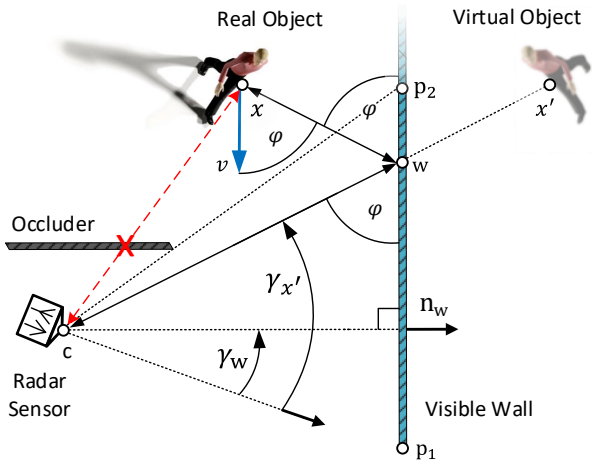


Figure 18: NLOS geometry and velocity estimation from indirect specular wall reflections. The hidden velocity $\mathbf{v}$ can be reconstructed from the radial velocity v_r by assuming that the road user moves parallel to the wall, i.e., on a road. The figure indicates all required angles and vectors.

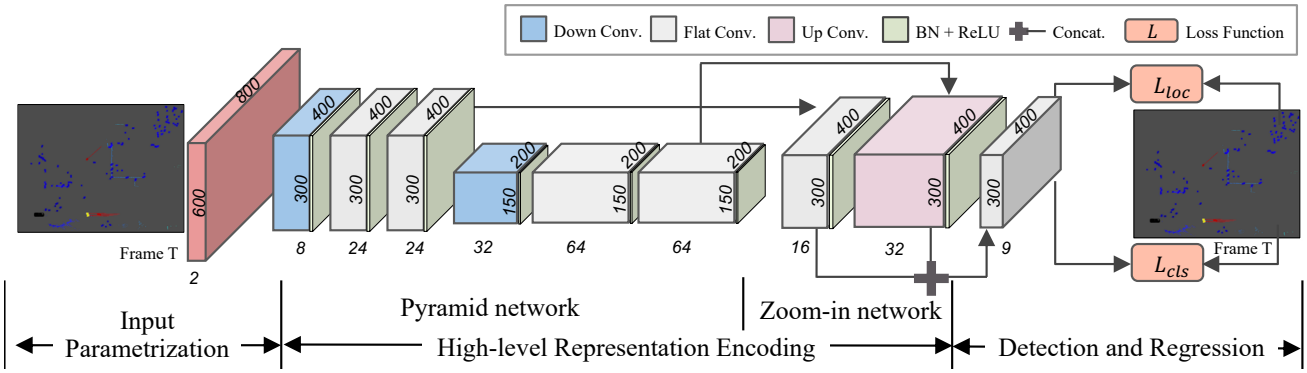


Figure 19: NLOS detection architecture. The network accepts the radar point cloud data at current frame T as input, and outputs predictions for frame T. The features are downsampled twice in the pyramid network, and then upsampled and concatenated by the zoom-in network.

To estimate the velocity $\mathbf{v}$ we need the normalized relay wall vector $\tilde{\mathbf{p}} = \mathbf{p} \cdot \|\mathbf{p}\|^{-1}$. Since we assume the object is moving parallel to the wall we derive $\mathbf{v} = s \cdot \tilde{\mathbf{p}}$ for some $s \in \mathbb{I}$. Since $\|\tilde{\mathbf{p}}\| = 1$ we know $|s| = \|\mathbf{v}\|$. Because the sign of s depends on the direction of $\tilde{\mathbf{p}}$ and therefore $\mathbf{p}$, we need a convention to determine $\mathbf{p}$. By convention we require $\mathbf{p} = \mathbf{p}_2 - \mathbf{p}_1$ to be rotated $\frac{\pi}{2}$ radians anti-clockwise to the wall normal $\mathbf{n_w}$ which was defined earlier. Intuitively this means $\mathbf{p}$ points to the left if the wall is in front of the sensor, this case is depicted in Fig. 18.

With this convention and $|s| = \|\mathbf{v}\| = \frac{|v_r|}{|\cos\varphi|}$ the remaining part is to determine the sign of s. This comes down to the sign of v_r and the angle between the normal $\mathbf{n_w}$ and the outgoing ray $\mathbf{x}' - \mathbf{c}$. We define $\gamma_{\mathbf{x}'}$ and $\gamma_{\mathbf{w}}$ as the angles of $\mathbf{x}' - \mathbf{c}$ and $\mathbf{n_w}$ relative to the sensor's coordinate system, again see Fig. 18. The sign of s is then given by

$$\operatorname{sgn}(s) = \operatorname{sgn}(v_r) \cdot \operatorname{sgn}(\gamma_{\mathbf{x}'} - \gamma_{\mathbf{w}}). \tag{47}$$

In the above equation (47) the sign of v_r distinguishes approaching and departing objects. Whereas, $\operatorname{sgn}(\gamma_{\mathbf{x}'} - \gamma_{\mathbf{w}})$ determines the object's allocation to the left or right half-plane with respect to the normal $\mathbf{n_w}$. Together, they encode the direction of the velocity $\mathbf{v}$. Putting everything together yields the estimated velocity

$$\mathbf{v} = \operatorname{sgn}(v_r) \cdot \operatorname{sgn}(\gamma_{\mathbf{x}'} - \gamma_{\mathbf{w}}) \cdot \frac{|v_r|}{|\cos\varphi|} \cdot \tilde{\mathbf{p}}. \tag{48}$$

This method of reconstructing a velocity vector from a radial velocity is not limited to the specific scenario we are considering here. This can be applied whenever we have strong believes about the direction an object is moving. For example we could obtain the road geometry from a map or by use of additional sensors.

5. Model Architectures & Experimental Settings

In this section, we explain the architecture and training details of the detection models and tracking models we used in Tab. 1 and Tab. 2 in the main paper.

5.1. Detection Models

The detection task is to estimate oriented 2D boxes for pedestrians and cyclists, given a BEV point cloud $\tilde{\mathbf{U}}$ as input.

5.1.1 Proposed Detection model

The overall proposed detection pipeline consists of three main stages: (1) input parameterization that converts a BEV point cloud to a sparse pseudo-image; (2) high-level representation encoding from the pseudo-image using a 2D convolutional backbone; and (3) 2D bounding box regression and detection with a detection head.

Fig. 19 shows the network architecture we used for detection. This model is mostly the same as the tracking model shown in Fig. 20 , except that there is only a single frame at timestamp T in both input and output ($n = 0$) and that consequently there is no feature map fusion across multiple frames between the pyramid network and the zoom-in network.

Input Parameterization We denote by $\mathbf{u}$ a d-dimensional ($d = 4$) point in a raw radar point cloud $\tilde{\mathbf{U}}$ with coordinates x, y (derived from the polar coordinates $\tilde{\phi}, \tilde{r}$), velocity $\tilde{v_r}$, and amplitude $\tilde{\sigma}$. We first preprocess the input by taking the logarithm

of the amplitude $\tilde{\sigma}$ to get an intensity measure $s = \log \tilde{\sigma}$. As a first step, the point cloud is discretized into an evenly spaced grid in the x-y plane, resulting in a pseudo-image of size $(d-2, H, W)$ where H and W indicate the height and width of the grid, respectively. Note that the final number of channels of the pseudo-image is $d-2$ instead of d, because the location information is encoded implicitly. Specifically, we mark each pixel that has points fallen into by putting intensity values of the points into that pixel, i.e., we do not explicitly encode the two location values of each point into the features of each pixel. We sum up the intensities from all points falling into the same pixel.

High-level Representation Encoding To efficiently encode high-level representations of the hidden detections, the backbone network contains two modules: a pyramid network and a zoom-in network. The pyramid network contains two consecutive stages to produce features at increasingly small spatial resolution. Each stage downsamples its input feature map by a factor of two using three 2D convolutional layers. Next, a zoom-in network upsamples and concatenates the two feature maps from the pyramid network. This zoom-in network performs transposed 2D convolutions with different strides. As a result, both upsampled features have the same size and are then concatenated to form the final output. All (transposed) convolutional layers use kernels of size 3 and are interlaced with BatchNorm and ReLU, as shown in Fig. 19.

Detection Head The detection head follows the setup of Single Shot Detector (SSD) [4] for 2D object detection. Specifically, each anchor predicts a 3-dimensional vector for classification (background / cyclist / pedestrian) and a 6-dimensional vector for bounding box regression (center, dimension, orientation, and velocity of the box).

Loss Functions Our overall objective function for training the proposed detection model contains a localization term and a classification term balanced by two weight factors α and β:

$$L = \alpha L_{loc} + \beta L_{cls}. \tag{49}$$

The localization loss is applied only to the frame T:

$$L_{loc} = L_{loc_T} = \sum_{u \in \{x,y,w,l,\theta,v\}} \alpha_u |\Delta u|, \tag{50}$$

where Δu is the localization regression residual between ground truth (gt) and anchors (a) defined by (x, y, w, l, θ):

$$\begin{aligned} \Delta x = x^{gt} - x^a, \quad \Delta y = y^{gt} - y^a, \quad \Delta v = v^{gt} - v^a, \\ \Delta w = \log \frac{w^{gt}}{w^a}, \quad \Delta l = \log \frac{l^{gt}}{l^a}, \quad \Delta \theta = \sin(\theta^{gt} - \theta^a). \end{aligned} \tag{51}$$

We do not distinguish the front and back of the object, therefore all θ's are within the range $[-\frac{\pi}{2}, \frac{\pi}{2})$. We also do not normalize Δx and Δy due to the noise of the data.

The classification loss is the focal loss L_{cls} from [4]:

$$L_{cls} = -\alpha(\mathbf{1} - \mathbf{p}^a)^\gamma \odot \log(\mathbf{p}^a) \odot \mathbf{y}^{gt}, \tag{52}$$

where $\odot$ is the element-wise product between 3-dimensional vectors. $\mathbf{1}$ is filled with ones, $\mathbf{p}^a$ is the anchor a's probability predictions (after applying *softmax* to the last layer's output) for the three classes, and $\mathbf{y}^{gt}$ is a one-hot vector encoding the true class label.

5.1.2 Baselines

SSD Baseline The input parameterization and detection head is the same as that of the proposed model. The backbone is a 3-layer convolutional neural network, with each layer followed by a ReLU layer. The (kernel size, stride, output channel number) settings for the three layers are: (3, 2, 8), (3, 1, 16), (3, 1, 32), respectively. This model has been trained on the same data set as the proposed model, with the manually optimized hyperparameters.

PointPillars Baseline We adopted a version for BEV point cloud detection from the original PointPillars for 3D detection. The input is 7-dimensional, including x, y, x_c, y_c, x_p, y_p, and s, where x, y are the 2D locations, the c subscript denotes distance to the arithmetic mean of all points in the pillar, the p subscript denotes the offset from the pillar x, y center, and s denotes the intensity, following the notations used in [2]. All the other settings are kept the same as the original model. This model has been trained on the same data set as the proposed model, with the manually optimized hyperparameters.

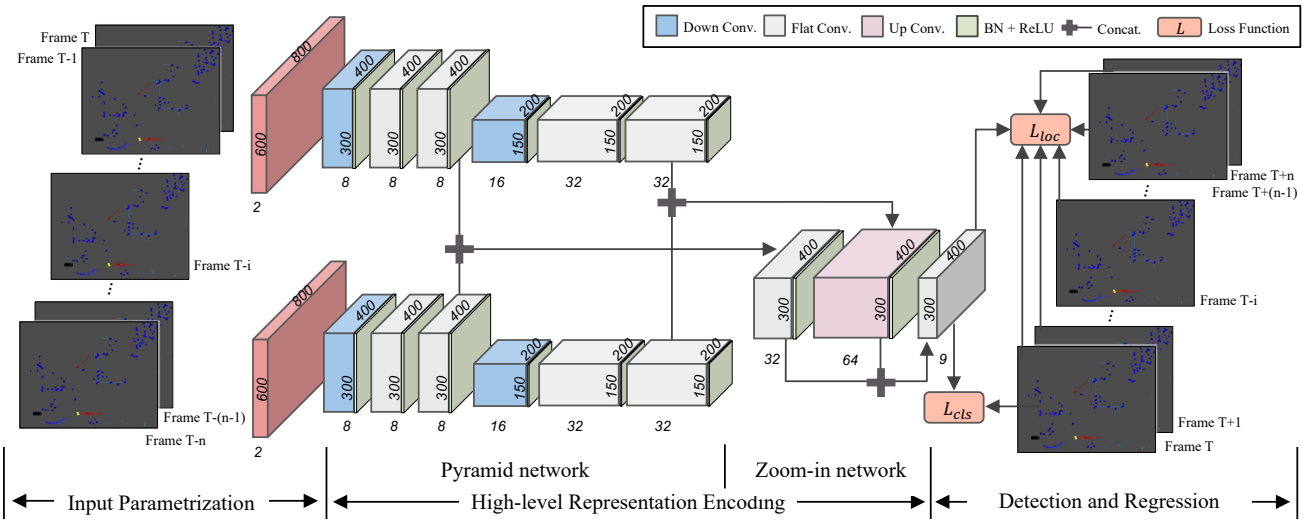


Figure 20: NLOS detection and tracking architecture. The network accepts the current frame T and the past n radar point cloud data as input, and outputs predictions for frame T and the following n frames. The features are downsampled twice in the pyramid network, and then upsampled and concatenated by the zoom-in network. We merge the features from different frames at both levels to encode high-level representation and fuse temporal information.

5.2. Tracking models

Our model jointly learns tracking with future frame prediction, inspired by Luo et al. [6]. At each timestamp, the input is from current and its n preceding frames, and predictions are for the current plus the following n future frames.

One of the main challenges is to fuse temporal information. A straightforward solution is to add another dimension and perform 3D convolutions over space and time. However, this approach is not memory-efficient and computationally expensive given the sparsity of the data. Alternatives can be early or late fusion as discussed in [6]. Both fusion schemes first process each frame individually, and then start to fuse all frames together.

Instead of such one-time fusion, our approach leverages the multi-scale backbone and performs fusion at different levels. Specifically, we first perform separate input parameterization and high-level representation encoding for each frame as described in Sec. 5.1.1. After the two stages of the pyramid network, we concatenate the $n+1$ feature maps along the channel dimension for each stage. This results in two feature maps of sizes $\big((n+1)C_1, \frac{H}{2}, \frac{W}{2}\big)$ and $\big((n+1)C_2, \frac{H}{4}, \frac{W}{4}\big)$, which are then given as inputs to the two upsampling modules of the zoom-in network, respectively. The rest of the model is the same as that of the detection model. By aggregating temporal information across $n+1$ frames at different scales, the model is allowed to capture both low-level per-frame details and high-level motion features. We refer to Fig. 20 for an illustration of our architecture.

Loss Functions Our overall objective function for tracking contains a localization term and a classification term

$$L = \alpha L_{loc} + \beta L_{cls}. \tag{53}$$

The localization loss is a sum of the weighted localization loss for the current frame T and n frames into the future:

$$L_{loc} = \sum_{t=T}^{T+n} \beta_t L_{loc_t} \quad \text{with} \quad L_{loc_t} = \sum_{u \in \{x,y,w,l,\theta,v\}} \alpha_u |\Delta u|, \tag{54}$$

where Δu is the localization regression residual between ground truth (gt) and anchors (a) defined by (x, y, w, l, θ):

$$\Delta x = x^{gt} - x^a, \quad \Delta y = y^{gt} - y^a, \quad \Delta v = v^{gt} - v^a,$$
$$\Delta w = \log \frac{w^{gt}}{w^a}, \quad \Delta l = \log \frac{l^{gt}}{l^a}, \quad \Delta \theta = \sin(\theta^{gt} - \theta^a). \tag{55}$$

We do not distinguish the front and back of the object, therefore all θ's are within the range $[-\frac{\pi}{2}, \frac{\pi}{2})$. For classification, we adopt the focal loss L_{cls} from [4]:

$$L_{cls} = -\alpha(\mathbf{1} - \mathbf{p}^a)^\gamma \odot \log(\mathbf{p}^a) \odot \mathbf{y}^{gt}, \tag{56}$$

where $\odot$ is the element-wise product between 3-dimensional vectors. $\mathbf{1}$ is filled with ones, $\mathbf{p}^a$ is the anchor a's probability predictions for the three classes, and $\mathbf{y}^{gt}$ is a one-hot vector encoding true class label.

5.3. Training Details

Data sets We split the original data set into non-overlapping training and validation sets, where the validation set consists of four scenes with 20 sequences and 3063 frames. For both training and validation, the region of interest is a large area of $60\,\mathrm{m} \times 80\,\mathrm{m}$. We use resolution $0.1\,\mathrm{m}$ to discretize both x, z axes into a 600×800 grid.

Input Parameterization For each tracking model used in the experiments, it takes the current and the preceding three frames as input and predicts the bounding box for the current and the three future frames, i.e., $n = 3$. For the model with velocity (v), the final feature map size after input parameterization is $(2, H, W)$ (both v and s are encoded); for the model without velocity (v), the final feature map size after input parameterization is $(1, H, W)$ (only s is encoded). For pixels that have more than one point fallen into after discretization, we average all the velocity values of the points within that pixel and sum up all the intensity values of the points within that pixel. We also experimented with averaging the intensity values, but find that the result is not improved. We think the reason may be that accumulated intensities also indicate the density of points in the grid, which helps with detection.

Detection Head For training, we assign each ground truth box to its highest overlapping predicted box. We only associate one bounding box with each feature map cell. And the associated bounding box is positive if and only if it is located at the same feature map cell where the ground truth box is located.

Loss Functions In our experiments, we set $n = 3$ for the tracking frame number, and we used $\alpha = 1, \beta = 1$ for the overall loss function:

$$L = \alpha L_{loc} + \beta L_{cls}. \tag{57}$$

The overall localization loss L_{loc} is a weighted sum of the 4 L_{loc_t}'s of the individual frames:

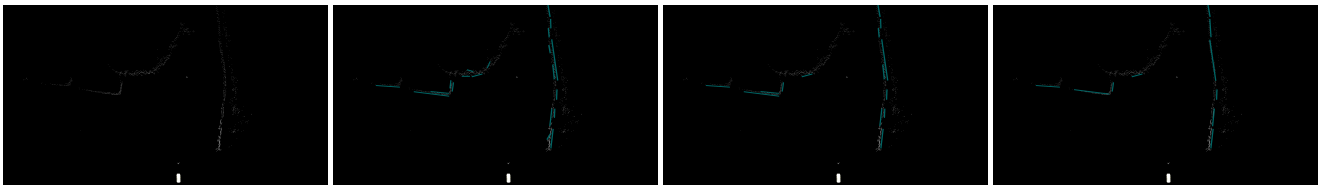
$$L_{loc} = L_{loc_T} + 0.75 L_{loc_{T+1}} + 0.5 L_{loc_{T+2}} + 0.25 L_{loc_{T+3}}. \tag{58}$$

The loss weight α_u for different localization parameters are $\alpha_x = \alpha_y = 0.75, \alpha_w = \alpha_l = 1, \alpha_\theta = 1.25, \alpha_v = 1$ for both detection and tracking models. For the classification loss, we set $\alpha = 0.99, \gamma = 2$ for the focal loss:

$$L_{cls} = -\alpha(\mathbf{1} - \mathbf{p}^a)^\gamma \odot \log(\mathbf{p}^a) \odot \mathbf{y}^{gt}, \tag{59}$$

We train from scratch using Adam optimizer to optimize L, with learning rate 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. We train on a single GPU for 30 epochs when it converges. The batch size is 32 and 4 for detection and tracking model, respectively.

We also report the performance of our tracking model in terms of speed during inference stage. We test the tracking model that, in one forward pass, simultaneously takes as input the current and preceding 3 frames and predicts for the current and future 3 frames. During inference, this unoptimized tracking model runs $40\,\mathrm{ms}$ on three consumer GPUs for one forward pass.



(a) Raw lidar BEV point cloud.

(b) Raw Detections. Detected line segments after using the LSD algorithm.

(c) Filtered Detections. Line segments after filtering those with length shorter than 1m in 21b.

(d) Filtered and Merged Detections. Final relay wall estimation result after applying the merging procedure to the overlapping parallel segments.

Figure 21: Visualizations of estimated relay walls at different filtering stages.

5.4. Relay Wall Estimation

We use first-response pulsed lidar measurements of a separate front-facing lidar sensor to recover the geometry of the visible wall. Specifically, we found that detecting line segments in a binarized binned BEV is robust using [15], where each bin with size $0.01\,\mathrm{m}$ is binarized with a threshold of 1 detection per bin. We observe that raw detections from the LSD algorithm contain two types of artifacts. The first artifact is the short line segments, which we filter out by using the threshold of $1\,\mathrm{m}$. The second artifacts are overlapping parallel segments that correspond to either a part or the whole of the same mirror[1], which we address by merging them into a big segment. The result of this process is shown in Fig. 21.

We show raw lidar BEV point clouds in Fig. 21a and detected line segments after using the LSD algorithm in Fig. 21b. Without the post-processing step, it can be observed that there are many short and overlapping segments in this raw detection. Fig. 21c shows the line segments after filtering those with length shorter than $1\,\mathrm{m}$. Finally, Fig. 21d shows the final relay wall estimation result after merging the overlapping parallel segments.

6. Additional Results

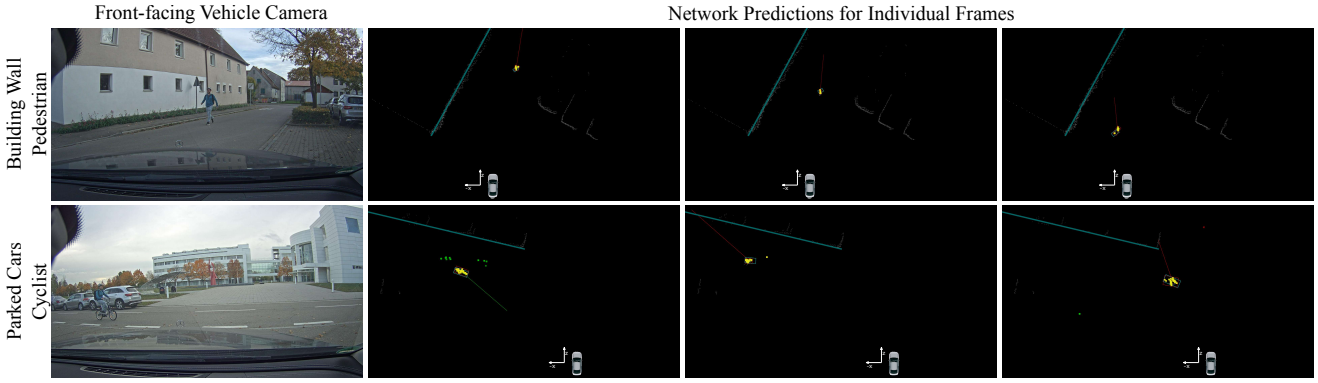


Figure 22: Additional joint detection and tracking results on LOS objects in training data set. Note that all the previous figures of joint detection and tracking results are for NLOS objects in testing data set. Here, as a sanity check, we also show the LOS object detection and tracking results on training data set, which is a much easier task than NLOS object prediction on testing data set.

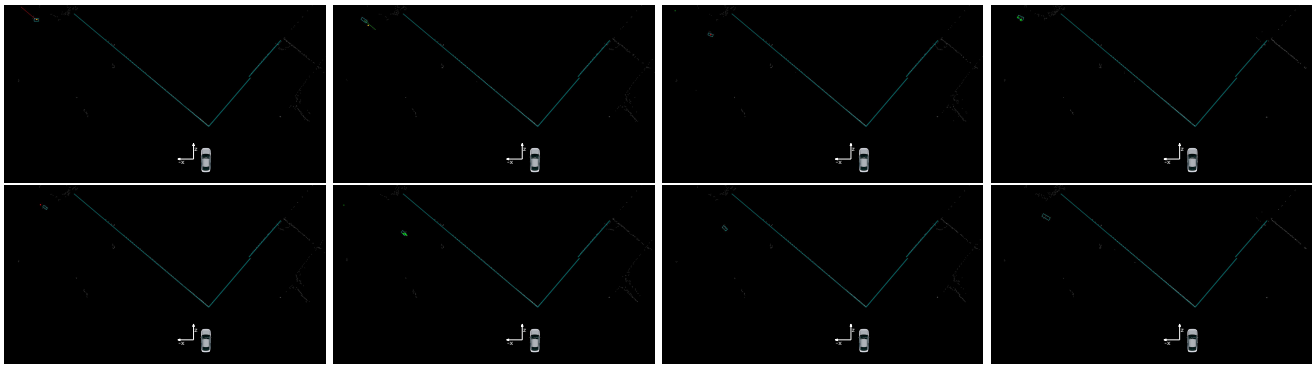


Figure 23: Additional joint detection and tracking results on NLOS objects in testing data set. Here, we show more results where a hidden object became visible by lidar but not radar. Due to noisy data in some of these frames, there are no labeled radar points. However, our model is able to detect correctly by reasoning about sequences of frames instead of a single one, validating the proposed joint detection and tracking approach.

[1] two parallel segments are overlapping if one segment contains a part of other segment's projection onto it

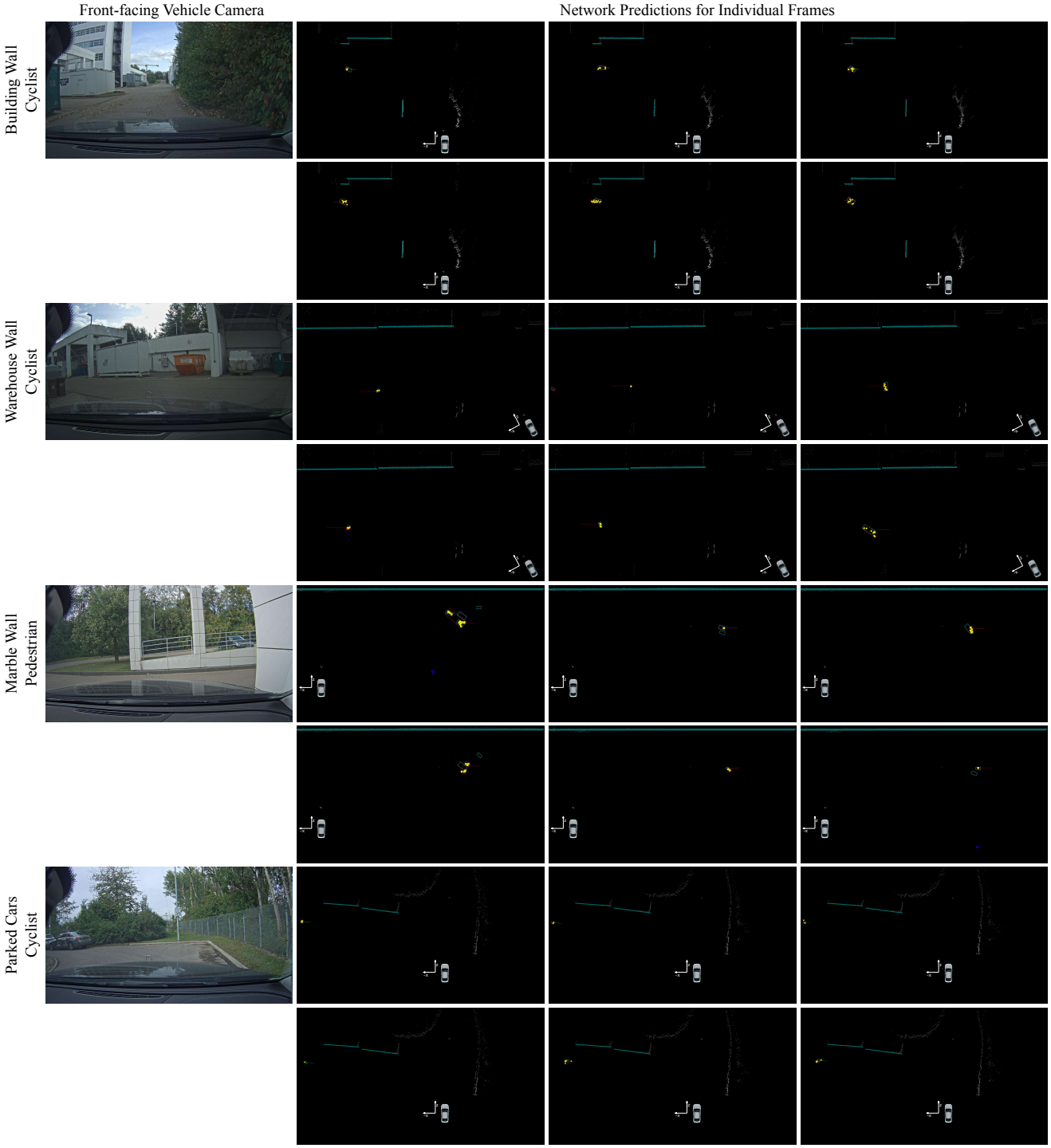


Figure 24: More joint detection and tracking results on testing data sets. The four scenes are the same as shown in the main paper. Here we show more frames where the NLOS objects are jointly detected and tracked by our model.

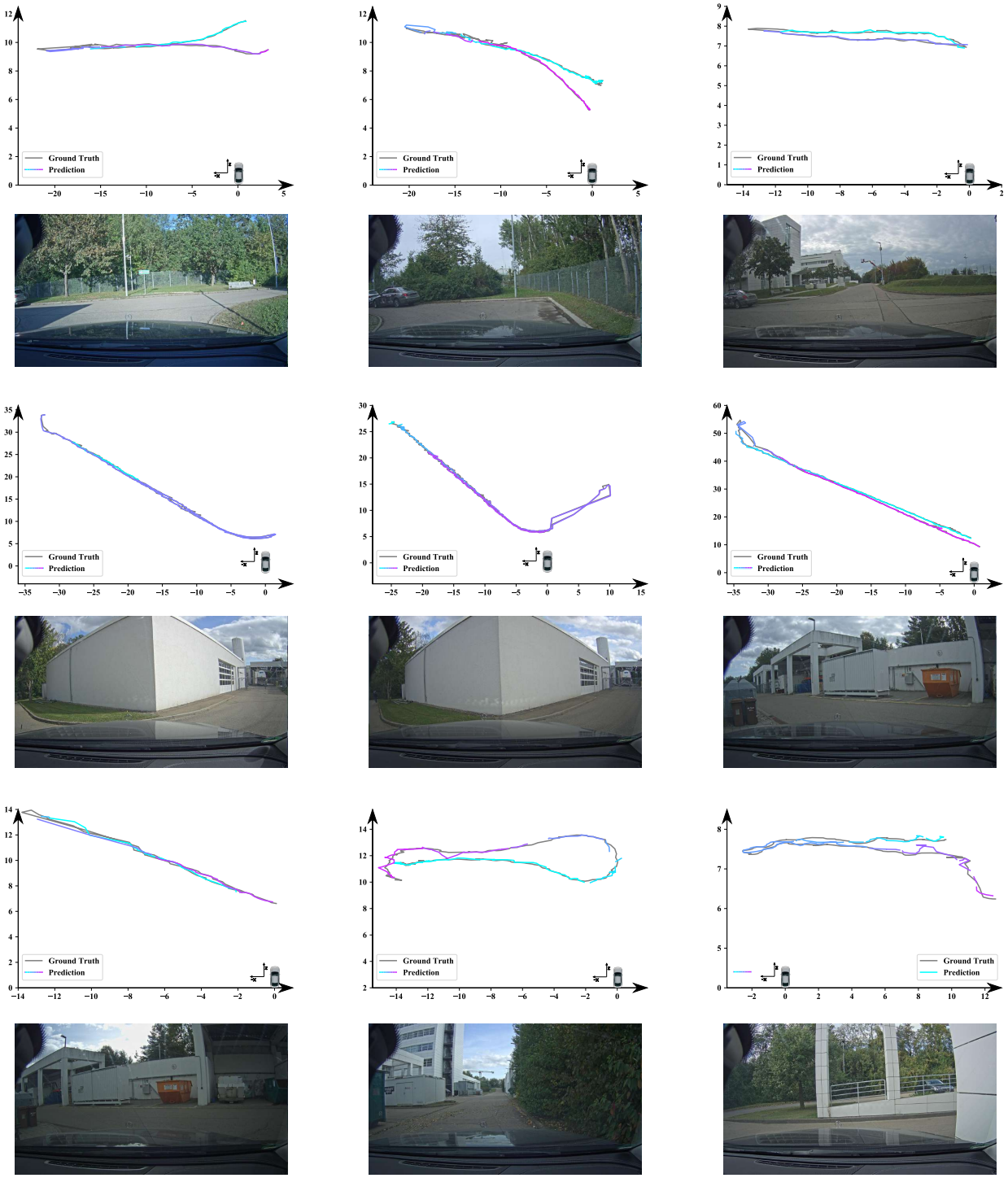


Figure 25: Tracking trajectories for both training and testing data sets. Here we show nine scenes in total. The top-middle scene and the last three scenes are from the testing data set. For each scene, the first row is the trajectory and the second row is the front-facing vehicle camera. We can see a variety of wall types, trajectories and viewpoints of the observing vehicles.

In Fig. 22, we show more joint detection and tracking results on NLOS objects in training data set. Note that all the other figures of joint detection and tracking results are for NLOS objects in testing data set. Here, as a sanity check, we show the LOS object detection and tracking results on training data set, which is a much easier task than NLOS object prediction on testing data set. In Fig. 23 and Fig. 24, we show more joint detection and tracking results on NLOS objects in testing data set. Specifically in Fig. 24, the four scenes are the same as in the main paper. Here we show more frames where the NLOS objects are jointly detected and tracked by our model. In Fig. 23, we show more results where a hidden object became visible by lidar but not radar. Due to noisy data in some of these frames, there are no labeled radar points. However, our model is able to correctly detect by reasoning about sequences of frames instead of a single one, validating the proposed joint detection and tracking approach. In Fig. 25, we show nine tracking trajectories for both training and testing data sets. We can see a variety of wall types, trajectories and viewpoints of the observing vehicles.

References

[1] W. Chen, S. Daneau, F. Mannan, and F. Heide. Steady-state non-line-of-sight imaging. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6790–6799, 2019. 7

[2] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 21

[3] J. Li and P. Stoica. *MIMO Radar Signal Processing*. John Wiley & Sons, Inc., Hoboken, New Jersey, USA, 2008. 10

[4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 21, 22

[5] D. B. Lindell, G. Wetzstein, and V. Koltun. Acoustic non-line-of-sight imaging. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6780–6789, 2019. 7

[6] W. Luo, B. Yang, and R. Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018. 22

[7] D. Lynch. *Introduction to RF stealth*. The SciTech radar and defense series. SciTech, 2004. 7

[8] S. M. Patole, M. Torlak, D. Wang, and M. Ali. Automotive Radars: A review of signal processing techniques. *IEEE Signal Processing Magazine*, 34(2):22–35, 2017. 5

[9] M. A. Richards, J. Scheer, W. A. Holm, and W. L. Melvin. *Principles of modern radar*. Citeseer, 2010. 7

[10] H. Rohling. Ordered Statistic CFAR Technique an Overview. In *2011 12th International Radar Symposium (IRS)*, pages 631–638, Leipzig, Germany, 2011. 12

[11] K. Sarabandi, E. S. Li, and A. Nashashibi. Modeling and measurements of scattering from road surfaces at millimeter-wave frequencies. *IEEE Transactions on Antennas and Propagation*, 45(11):1679–1688, 1997. 7

[12] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick. A Multi-Stage Clustering Framework for Automotive Radar Data. In *IEEE 22nd Intelligent Transportation Systems Conference (ITSC)*, Auckland, New Zealand, 2019. IEEE. 15

[13] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick. Automated Ground Truth Estimation of Vulnerable Road Users in Automotive Radar Data Using GNSS. In *IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pages 5–9, Detroit, MI, USA, 2019. IEEE. 2

[14] N. Scheiner, S. Haag, N. Appenrodt, B. Duraisamy, J. Dickmann, M. Fritzsche, and B. Sick. Automated Ground Truth Estimation For Automotive Radar Tracking Applications With Portable GNSS And IMU Devices. In *20th International Radar Symposium (IRS)*, pages 1–10, Ulm, Germany, 2019. 2

[15] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: a line segment detector. *Image Processing On Line*, 2:35–55, 2012. 24