Conf. ID: 1570870012

Simultaneous AMCW ToF Camera and FMCW Radar Simulation

Harald Gietler[1], Barnaba Ubezio*[1] and Hubert Zangl

2023 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) | 978-1-6654-5383-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/I2MTC53148.2023.10176005

Abstract— **As two of the main sensors utilized in robotics, environment representation and navigation, ToF cameras and radars are often combined in sensor fusion frameworks. Realistic and complete simulation plays a crucial role in fast prototyping, performance assessment, and model analysis of systems relying on these sensing technologies. Typical simulation environments, however, don't model key characteristics, such as the radar sensor's time domain raw data, antenna arrays' structure, or parasitic effects of depth estimation of ToF cameras. This article presents a simultaneous ToF camera and radar simulation based on the Unity 3D engine. The two sensors are modeled from a single custom RGB camera object and are inherently time synchronized. The simulation provides ToF depth and intensity estimates, together with time domain radar data and realistic detrimental effects, allowing testing of standard post-processing algorithms. Several parameters can be configured, such as field of view, modulation characteristics, and the number and position of antennas. Moreover, simulation objects can have different material properties for the two sensors making them more or less visible for each sensor. The approach focuses on user-friendly, realistic sensor simulation while keeping computational complexity within acceptable limits to ensure applicability. The capabilities of the proposed simulator are validated by comparing real and simulated sensors in different scenarios.**

I. INTRODUCTION

To generate a spatial representation of the environment, Amplitude Modulated Continuous Wave (AMCW) Time-of-Flight (ToF) cameras and Frequency Modulated Continuous Wave (FMCW) radars have been employed in applications such as human tracking [1], [2], robotics [3], [4], mapping and navigation [5], [6].

ToF cameras use infrared light to provide dense point clouds from depth estimates in one shot at a high frame rate. FMCW radars generally have lower spatial resolution and frame rate than AMCW ToFs. Still, they provide direct velocity estimates and work under harsh environmental conditions, such as bad illumination, rain, dust, or smoke. Because of their common principle of measuring the echo from target objects, ToFs and radars suffer from common error sources such as multipath phenomena. Both sensors are generally compact and lightweight, but their response to the environment is generally different due to their fundamentally different excitation mode. Consequently, complementary usage of such technologies in sensor fusion applications is proposed [7]. The increasing complexity of tasks in the application domains requires simulation-based pre-testing and data analysis. In [8], Cheng et al. investigate the generation and enhancement of radar point clouds by proposing a data-driven detector trained with Lidar point clouds in a supervised learning approach. All the data for both sensors are collected in real life, which is not always feasible. Steinbaeck et al. [9] further investigated sensor fusion frameworks for ToF cameras and radars for occupancy grid mapping and navigation, based on merging point clouds from the two sensors. A joint simulation would allow speeding up data collection, testing configuration changes, and generating training data for learning algorithms. [10] also describes a fusion framework with a AMCW ToF and a 60 GHz FMCW radar for human tracking in indoor environments. Performances are closely related to the post-processing algorithms used for building Range-Angle and Range-Doppler maps with the radar. In a simulation, it would not be possible to test the different algorithms without having access to the time domain raw data. Hence, the need to provide such measurements, namely the mixed signals comprising transmitted and received waveforms embedded in noise.

In the context of radar simulation, some frameworks have been proposed [11], but they lack the aforementioned features. Recently, the authors in [12] implemented an FMCW radar simulation for robotics applications, where time domain data are provided. Limitations of this work include very low resolution, absence of noise, and an approximation of the multiple antennas model, which only adds pre-computed phase shifts to a single signal. This ignores important phenomena such as the geometry-dependent intersection of a physical antenna array. In the context of ToF, [13], and the authors in [14] implemented real-time realistic simulations, which provide intensity and depth estimates, including parasitic effects.

The quality of the simulation depends on the quality of the employed models. A lack of realistic sensor models decreases simulation quality and the subsequent transfer of the findings from simulation to reality. We build upon [14], and [12] and propose a Unity 3D-based simultaneous AMCW ToF and FMCW radar simulation. With respect to the aforementioned works and state-of-the-art, the simulation proposed in this paper has the following characteristics:

1) Synchronization: the measurements of both sensors are generated in one shot from the same RGB camera

The authors are with the Institute of Smart Systems Technologies, University of Klagenfurt, 9020 Klagenfurt, Austria

[1]: Harald Gietler and Barnaba Ubezio contributed equally to this work as first authors.

*Corresponding author: barnaba.ubezio@aau.at

This work is partially funded by the European Leadership Joint Undertaking (ECSEL JU), under grant agreement No. 876019, within the ADACORSA project - Airborne Data Collection on Resilient System Architectures, as well as by the Austrian Ministry of Climate Action and Energy (BMK) under the grant agreement No. 880057 (CARNIVAL).

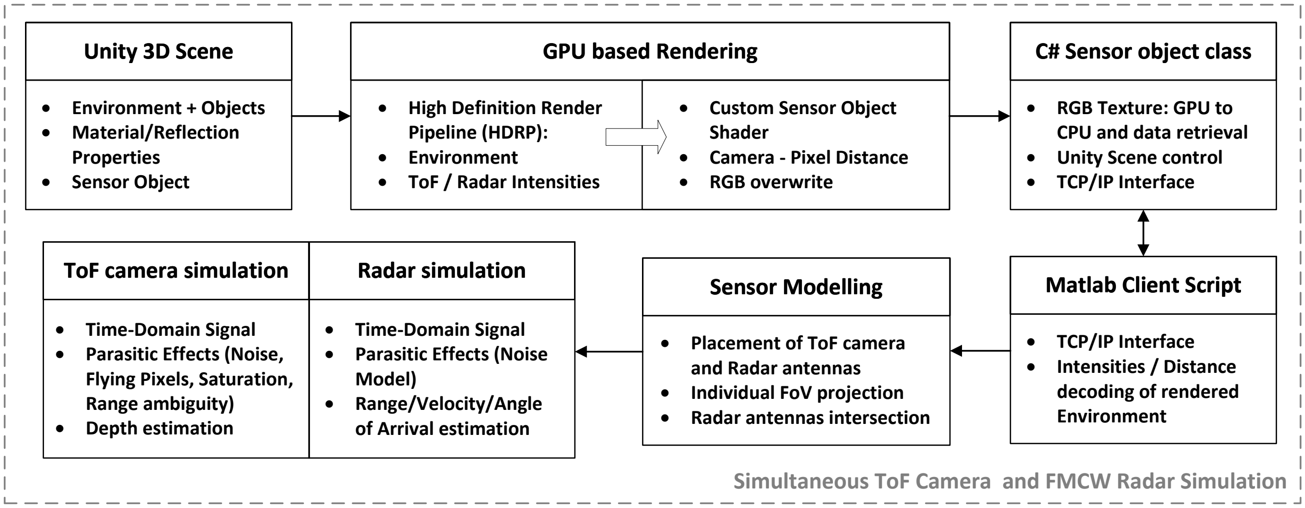
Unity 3D Scene
Environment + Objects
Material/Reflection Properties
Sensor Object
GPU based Rendering
High Definition Render Pipeline (HDRP):
Environment
ToF / Radar Intensities
Custom Sensor Object Shader
Camera - Pixel Distance
RGB overwrite
C# Sensor object class
RGB Texture: GPU to CPU and data retrieval
Unity Scene control
TCP/IP Interface
ToF camera simulation
Time-Domain Signal
Parasitic Effects (Noise, Flying Pixels, Saturation, Range ambiguity)
Depth estimation
Radar simulation
Time-Domain Signal
Parasitic Effects (Noise Model)
Range/Velocity/Angle of Arrival estimation
Sensor Modelling
Placement of ToF camera and Radar antennas
Individual FoV projection
Radar antennas intersection
Matlab Client Script
TCP/IP Interface
Intensities / Distance decoding of rendered Environment
Simultaneous ToF Camera and FMCW Radar Simulation

Fig. 1: An overview of the simulation pipeline to generate realistic sensor data.

image and have the same timestamp.

2) Independence: by exploiting independent color channels and illumination layers, targets can be made more or less visible for one sensor independent of the other to model the different real-world reflectivity properties.
3) Time domain radar data: the simulated radar's output is the mixed signal from transmitter and receiver antennas, with added noise floor model.
4) Radar antenna configuration: the user can physically add receivers to build an antenna array, choosing the position of every element without the need to re-rendering the camera images.

These characteristics allow for efficient modeling of the ToF and the radar simultaneously, change configuration parameters and generate synchronous and realistic point clouds. In the following, we will describe the concept architecture, revisit the measurement model used to generate simulated data, and compare the obtained results with real-world data.

II. Simulator Concept

The proposed simulation concept of this paper provides a user-friendly and accurate platform for the simultaneous simulation of FMCW radars and AMCW ToF cameras. An overview of the simulation framework components and their interconnections is shown in Fig. 1. In the following, the individual features of the simulator are rigorously explained.

A. Unity 3D Scene

The simulation scenery is modeled in Unity 3D [15], providing a high degree of usability. The Unity 3D developer scene offers interactive 3D content/libraries with various environmental objects and material distributions. ToF and radar functionality are added using a custom-made sensor object comprised of a camera and two light sources. The independent light sources represent the ToF Infra-Red (IR) illuminator and the radar transmitter (Tx) antenna. The illumination response of the environment to the camera object depends on the domain topology and used material with properties such as ”metallic” and ”smoothness”. Layering techniques are used to create independent responses of a single object to the two light sources enabling independence and physical difference of the radar and ToF signals. Assigned material (e.g., specularity, diffuse response) properties define how object surfaces reflect incoming ToF and radar signals. Note that the camera and the light sources have a common pose and the light sources use different colors.

B. GPU based Rendering, C# Sensor class, Matlab Client Script

The Graphics Processing Unit (GPU)-based rendering of the camera image as a function of the light source and the scenery is performed by Unity 3D's High Definition Render Pipeline (HDRP). HDRP is a customizable and scriptable render pipeline that supports physically based lighting techniques. We use a Bidirectional Reflectance Distribution Function (BRDF), comprised of the isotropic multi-scattering GGX function for the specular component [16], and Disney Diffuse for the diffuse component [17]. The rendered 3-channel (32-bit each) image contains the RGB intensities; one channel is used by the radar receiver (Rx), assumed coincident with the Tx, and one is used by the ToF illuminator. A custom shader script retrieves the GPU z-buffer data, calculates the distance to each pixel in the scene, and copies the depth image to the third color channel. Afterward, the image is transferred to the Central Processing Unit (CPU) using a C# script and published as TCP/IP stream. The GPU-CPU architecture of the framework is asynchronous, yielding an optimized update rate independent of the individual computational complexity. The C# script additionally controls the settings and objects' motion in the Unity 3D scene, which is useful for testing velocity estimation with the radar. A Matlab script receives and interprets the intensity and depth images.

C. Sensor Modelling

The rendered image has a resolution of $W \times H = 1000 \times 342$ pixels and a Field of View (FOV) of $F^h \times F^v = 100° \times 45°$. These are

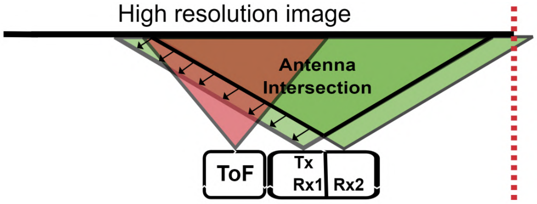

Fig. 2: 2D visualization of the simultaneous multiple sensors modeling from a single image. If the FOV of any sensor/antenna exceeds the limits of the high-resolution image (red dotted line), the relative pixels are discarded.

adjustable numbers which, in this case, exceed the resolution and FOV of common ToF cameras and radars. Therefore, the image is cut according to the region effectively covered by each sensor. This approach is practically used to model multiple antennas of the radar sensors or multiple ToF cameras without re-rendering.

Fig. 2 shows a 2D representation of a possible alignment of a ToF camera and a radar comprised of two Rx antennas. With homogeneous transformations, we can specify the rotation and translation of each element of the sensor system to the RGB camera origin. The outputs of this block are distance and intensity maps of the cut and transformed visible regions. The procedure to calculate them is described in the following.

The pixel-wise, 3D X and Y coordinates are first computed for the RGB image (the Z is already given by the depth information) by rearranging the central perspective model equations [18]:

$$u = f_x \frac{X}{Z}, v = f_y \frac{Y}{Z} \tag{1}$$

$$X = \frac{Zu}{f_x} = \frac{Z(u/X_{max})}{(W/2)} \tag{2}$$

$$Y = \frac{Z(v/Y_{max})}{(H/2)} \tag{3}$$

where f_x, f_y are the focal lengths, $\{u, v\}$ the normalized pixel coordinates and $\{X_{max}, Y_{max}\} = tan(\{F^h, F^v\}/2)$ the maximum measurable coordinates. If, for one pixel, $\boldsymbol{X}_{rgb} = [X, Y, Z, 1]^T$, we can write for the generic sensor element s:

$$\boldsymbol{X}_s = \begin{bmatrix} R_s^{rgb} & t_s^{rgb} \\ 0_3 & 1 \end{bmatrix} \boldsymbol{X}_{rgb} \tag{4}$$

where R_s^{rgb} and t_s^{rgb} are the rotation matrix and translation vector representing the transformation between s and the camera origin. In the simulation, we consider the ToF origin coincident with the RGB camera one, so that $\boldsymbol{X}_{ToF} = \boldsymbol{X}_{rgb}$, and only the cut has to be performed. Then, as represented in Fig. 2, Tx/Rx1 and Rx2 are added by computing the respective measured coordinates with (4), substituting the transformation values according to mounting and antenna spacing. Such an approach also allows the simulation of standard radar Rx configurations with multiple antennas, e.g., linear, rectangular, or circular arrays.

For both ToF and radar, the visible regions according to their FOV are then cut out of the high-resolution image. To achieve this, we compute the pixel coordinates $\boldsymbol{x}_s$ with the camera matrix P:

$$\boldsymbol{x}_s = P\boldsymbol{X}_{rgb} = K_s[R_s^{rgb}|t_s^{rgb}]\boldsymbol{X}_{rgb} \tag{5}$$

where K_s is the known intrinsic parameter matrix, whose entries depend on the FOV of s and on the image resolution. This yields pixel values that are normalized such that the values below 0 and above $\{W, H\}$ correspond to the regions outside the sensor's FOV, and are therefore discarded. The cut is applied to the intensity and distance matrices, where the distance to each pixel is computed as $d_s = \sqrt{X_s^2 + Y_s^2 + Z_s^2}$.

Additionally, for the radar, we consider the intersection region between antennas as the visible region since, in real radars, each Rx antenna can only receive a waveform that was transmitted within the Tx FOV. The distance values from the intersection region are projected back to Tx/Rx1 to efficiently simulate the pixel-wise phase difference due to the antenna spacing. With this modeling, users can perform Angle of Arrival (AoA) estimation from the raw time domain data generated from the aligned distances. The high resolution allows for spacing between antennas in the millimeter order, as in the case of mmWave FMCW radars.

D. ToF camera simulation

AMCW ToF cameras transmit periodic light signals, which are reflected by the environment. The incoming signal is correlated with the outgoing one to estimate the phase shift caused by light's travel time and, ultimately, the distance traveled. By modeling the transmitted light as $g(t) = cos(\omega_T t)$ and the incoming reflection as $f(t) = A_T cos(\omega_T t - \phi_T)$, where ϕ_T and A_T denote the phase shift between the signals and the amplitude of the received signal, the correlation function evaluates to [14]:

$$C = \frac{A_T}{2} cos(\phi_T + \omega_T \tau). \tag{6}$$

The values for A_T and ϕ_T origin from the intensity and distance maps calculated before, where $\phi_T = 2\pi d_{ToF}/(\lambda/2)$, where $\lambda = c/f_T$ is the wavelength of the modulation frequency $f_T = 2\pi\omega_T$, and c is the speed of light. By selecting four observation phases as $\omega_0 \tau = i\frac{\pi}{2}$ for $i = 0...3$ yielding $C_0...C_3$, a suitable reconstruction algorithm for ϕ_T can be and a corresponding output distance estimator by using the estimated phase shift are given by

$$\hat{\phi_T} = atan\left(\frac{C_3 - C_1}{C_0 - C_2}\right), \quad \hat{d_{tof}} = \frac{c\hat{\phi_T}}{2\omega_0}. \tag{7}$$

The distance estimation procedure is performed pixel-wise and expanded to multiple parallel pixels yielding a depth image. Additional effects modeled in the ToF simulation are the following:

- Maximum unambiguous distance estimation due to the periodic nature of the received signal. If the distance to an object exceeds the limit, the estimate is mapped into the $\frac{\lambda_T}{2}$ range.

- The Flying Pixel effect occurs when the sensor pixel projects to an area containing depth discontinuities, e.g., a sharp edge. Thanks to the high-resolution image, which is 4× higher than the ToF one, a contribution of 4 pixels corresponds to a single sensor pixel signal.
- Cross-talk effect due to the tunnel effect in complementary metal-oxide-semiconductor (CMOS) technology [19]. The pixels of ToF cameras are grid-type positioned, so this effect is modeled using a radially symmetric Gauss Filter as isotropic approximation.

E. Radar simulation

In FMCW radars, a sinusoidal waveform with varying frequency is transmitted for a time T_c. Linear sweeps of the frequency bandwidth B are known as chirps. The transmitted signal can be represented as [20]:

$$x_T(t) = cos(2\pi f_c t + \pi K t^2) \tag{8}$$

where f_c is the lower frequency of the signal and $K = B/T_c$ is the slope rate of the chirp. The waveform is reflected by the target and captured by one or more receiving antennas after a time delay $\tau = 2(R + vt)/c$, where R and v are the range and velocity of the target, respectively. For Doppler velocity estimation, a number N_c of chirps can be transmitted in a single measurement frame. The received signal is mixed with the transmitted one and low-pass filtered, to provide so called Intermediate Frequency signal, whose real part is:

$$x_{IF}(t) = A_R cos(2\pi f_{IF} t + \phi_{IF}) \tag{9}$$

where A_R is the signal amplitude, $f_{IF} = K\tau$ is the constant beat frequency corresponding to the difference between transmitted and received waveforms, and $\phi_{IF} = 2\pi f_c \tau - \pi K \tau^2$ is the mixed signal phase. Finally, the mixed signal is sampled N_s times at the ADC frequency f_s to provide the raw time domain data which we are interested to model. Radars allowing low level access can be configured by varying N_s, N_c, f_s, B and T_c. These parameters are chosen on the basis of needed maximum range, velocity and resolution. Our simulation framework allows the configuration of all these quantities. More detailed information on FMCW radar theory and signal processing methods can be found in [21]. The generation of the time domain signal is summarized in the following. The mixed signal is first evaluated at each pixel, by discretizing the IF equations for all samples and chirps. For one pixel of coordinates u, v, we have:

$$\tau[n]_{u,v} = 2(R_{u,v} + v_{u,v} n) n_c / c \tag{10}$$

$$f[n]_{u,v} = 2\pi f_c \tau[n]_{u,v} + 2\pi K \tau[n]_{u,v} n - \pi K \tau[n]^2_{u,v} \tag{11}$$

$$x_{IF}[n]_{u,v} = A_R cos(f[n]_{u,v}) \tag{12}$$

where $1 \leq n_c \leq N_c$ is the chirps index, $n = n_s/f_s$, with $1 \leq n_s \leq N_s$ is the samples index, and A_R is the pixel intensity. The pixel's velocity v is computed from the difference between radial distance values R^k and R^{k-1} obtained at two consecutive steps k and $k-1$, with Δt as simulation rate. Notice that the values of R are simply given by the distance maps from the previous block. The pixels' contributions are summed up, and everything is repeated for each simulated antenna, obtaining the raw time domain signal x_{IF}, reshaped into a N_c x N_s x N_a matrix, where N_a is the number of Rx antennas.

TABLE I: Relevant parameters of the used sensors.

	PMD CamBoard ToF	Infineon BGT60TR13C
Field of View	$62° \times 45°$	$120° \times 60°$
Wavelength	850 nm	5 mm
Resolution	224×171	-
Modulation Frequency	20 MHz	60 GHz
Chirp Bandwidth	-	6 GHz
Sampling frequency	-	2 MHz
Number of samples	-	512
Number of chirps	-	1 (exp.1) - 16 (exp.2)
Number of Rx antennas	-	2
Unambiguous Range	7.5 m	$N_s c/(4B) = 6.4$ m

Radar tracking performances are greatly dependent on the target detection algorithm, related to the peaks in the Range-Fast Fourier Transform (FFT) spectrum. In real sensors, low-amplitude peaks corresponding to targets with low Radar Cross Section (RCS) might not exceed the noise floor level and therefore will not be detected. The proposed simulation takes this effect into account. Noise is usually modeled at the receiver level [22] and is a mixture of thermal noise, phase noise and other effects, the sum of which is approximated as additive white Gaussian noise. Having modeled the mixed signal, we add such WGN samples to it and obtain our final time domain radar signal $x_R[n] = x_{IF}[n] + w[n]$. The variance of $w[n]$ can be adjusted by the user, as its value varies with each device and in different conditions.

The obtained samples of x_R can be processed with standard radar signal processing algorithms for target tracking. We use 2D-FFT paired with an OS-CFAR detector and monopulse phase difference estimation to compute radial distance (range), velocity and AoA, from which the radar point cloud is generated. Refer to Section III for the one-to-one comparison with a real sensor.

III. Experimental Results

The commercial sensors used to validate the results in the real world are the PMD CamBoard pico flexx ToF camera and the Infineon BGT60TR13C 60GHz FMCW radar. The relevant specifications for the following experiments are summarized in Tab. I. All the parameters are configurable in the simulation. Notice that the radar's FOV is higher than the RGB image one, but these are theoretical values that are, in practice, significantly decreased, especially in presence of low reflectivity targets.

In the following, we describe two experiments conducted to test the performances of the proposed simulation.

A. Experiment I - Static Scene

In the following experiment, the simulation framework is qualitatively tested in the environment shown in Fig. 3, where multiple objects and materials are present, including metal, paper and concrete. The scene model in Unity 3D contains the most important items. Since the modelling

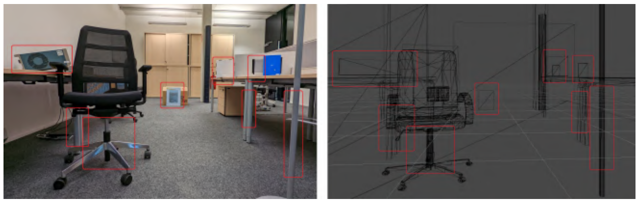

Fig. 3: Real-world and simulated scene with the most dominant targets highlighted.

effort was kept within certain limits, there is a visible gap between simulation and reality. For instance, the furniture in the simulation is taken from an available library and not modelled specifically. However, the key idea of this concept is not recreating a certain scenery in Unity 3D but rather provide realistic simulated sensor data. To each object in the simulated environment we assign materials and metallic/smoothness properties. By exploiting the light layer structure in Unity, or the independent color channels of the rendered image, surfaces can be made highly radar/ToF-reflective, transparent just for one sensor or other similar characteristics. To test such properties, we place a very good near-IR reflector on the right desk (next to the blue book), and check its visibility from the two sensors. Similarly, a strong radar corner reflector is placed behind a white paper box, which is made transparent for the radar but not for the ToF. In the case of the ToF camera, the intensity of the reflected light together with the scene depth information is processed yielding the distance estimate shown in the left image of Fig. 4. The corresponding real-wold experiment is shown in the right image of Fig. 4. It can be observed that the placed IR reflector dominates the intensity image. Note that the intensity at the neighboring pixel is also affected due to the cross-talk effect in CMOS technology. Due to the high intensity, the affected sensor pixels are saturating, leading to failing depth estimation. This effect can be seen in the corresponding real-world as well as simulated depth image shown in Fig. 4, where the estimated depth in the region of the reflector is zero and the neighboring region shows wrong distance estimates which are in turn the distance to the reflector. In this experiment, the response of different materials to the near-IR illumination can be observed in the intensity and corresponding depth images. The chair's backrest generates minimal reflections and is, therefore, not visible in either of the images. Contrary, the seating reflects the incoming illumination well and is accurately represented in the depth image, as well as the other metallic materials such as the aluminum plate in the back. Even though the camera simulation concept includes the most significant effects of standard ToF cameras, the compensation of minor discrepancies will require additional functional modeling. This is supported by the flexibility of the simulation concept, where additional functionalities can be added at any stage of the simulation pipeline.

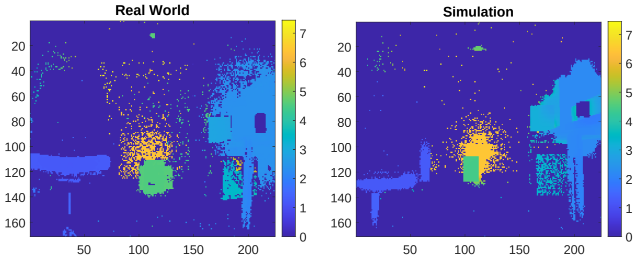

Fig. 4: A comparison between real world and simulated ToF camera data shows the importance of the modelled parasitic phenomena.

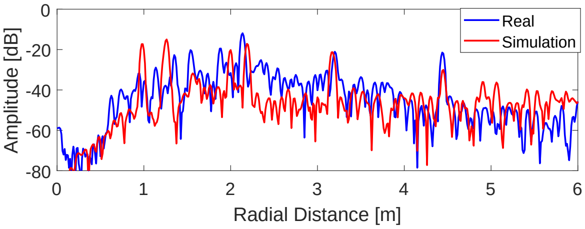

Fig. 5: Spectrum of the Range-FFT for the office scenario. Both signals contain many peaks, due to the complexity of the scene. The difference between real and simulated distances, in the order of cm for some of the peaks, are due to the fact that the scene in Unity is not an exact replica of the real environment. We also notice the similarity of the effect of a low pass filter (for DC leakage component removal) and the presence of the noise floor model in the simulation data.

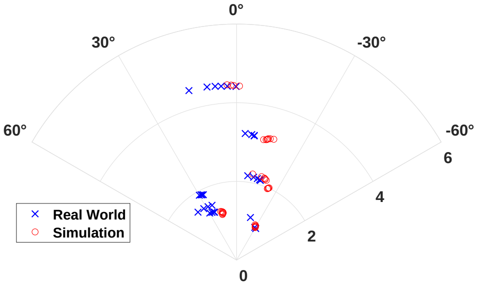

Fig. 6: Range-Angle map for the office scenario. With respect to the Range-FFT, we see that the CFAR detector algorithm only considers some of the reflections as valid targets, resulting in a strong similarity of the computed point clouds. Small spatial differences between real and simulated targets could be reduced with a better Unity model, the building of which is outside the scope of the paper.

The radar results are reported in Fig. 5 and Fig. 6. From the Range-FFT spectrum in Fig. 5 we notice the similar behavior of the two signals, even though the real radar sees additional reflections, e.g., in the region between 2.3 and 3 m. The object on the left desk, behind the chair just below 2 m of distance is also not visible in the simulation. The radar corner reflector at 3.2 m correctly shows a strong amplitude, making it clearly visible in both signals. In Fig. 6 we see the capabilities of the simulation to produce a point cloud analog to the real one, opening the possibility of generating such data for learning algorithms and testing purposes. The results have been obtained using the same configuration parameters, in real and simulated world, for all the signal processing

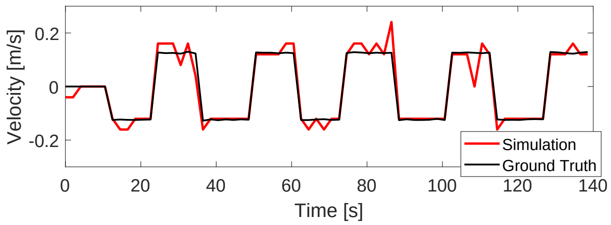

Fig. 7: Velocity plot computed from the Doppler-FFT on the simulated data for single target tracking.

algorithms mentioned in Section II-E. The simulation outputs the synchronized ToF images and radar measurements at 4-6 Hz according to the configuration.

B. Experiment II - Doppler velocity estimation

The second experiment is performed in simulation to test the effect of the Doppler-FFT on the generated radar raw data. For simplicity, we report the result of a single target tracking scenario, where the camera object in Unity is programmed to perform a simple back and forth motion while tracking the target. The radar is configured to send 16 chirps and the Doppler-FFT is computed by zero-padding the raw data along the chirp dimension, after pre-multiplying them with a Chebyshev window to enhance the main peak in the spectrum. In Fig. 7 are reported the results with respect to the ground truth (computed directly in Unity). We can notice some spikes during the constant velocity phases, reflecting the presence of noise in the raw time domain data. Proper configuration parameters for velocity computation are highly dependant on the scenario needed by the user. Our simulator allows to change the same parameters as the ones present on real radars, and therefore investigate which settings are best suited for more complex scenarios.

IV. CONCLUSIONS

ToF cameras and FMCW radars are already widely used in robotics for applications such as environment representation, obstacle avoidance, and navigation. Due to their characteristics they are often used in conjunction yielding improved perception of the environment. This paper introduces a real-time simulation concept that makes digital twins of such sensors already available in the design process of the target application. The simulation pipeline covers significant effects of commercial ToF cameras and FMCW radars. The framework is flexible and several parameters can be configured to model physical sensors, such as field of view, modulation characteristics, and the number and position of antennas. To keep the computational complexity in certain limits, the framework renders the scene once while using light layering and color distinguishing techniques to model the ToF camera and radar independently and allow for coherent data generation. This is required, because common materials have different reflection responses to the fundamentally different excitation frequencies of radar and ToF cameras. The capabilities of the proposed simulator are validated by comparing real and simulated sensors in a realistic office room scenario containing multiple physical objects.

REFERENCES

[1] M. Zhao, T. Li, M. Abu Alsheikh, Y. Tian, H. Zhao, A. Torralba, and D. Katabi, "Through-wall human pose estimation using radio signals," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7356–7365.

[2] T. M. A. Zulcaffle, F. Kurugollu, D. Crookes, A. Bouridane, and M. Farid, "Frontal view gait recognition with fusion of depth features from a time of flight camera," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 1067–1082, 2019.

[3] M. Zlatanski, P. Sommer, F. Zurfluh, and G. L. Madonna, "Radar sensor for fenceless machine guarding and collaborative robotics," in *2018 IEEE International Conference on Intelligence and Safety for Robotics (ISR)*. IEEE, 2018, pp. 19–25.

[4] B. Ubezio, C. Schöffmann, L. Wohlhart, S. Mülbacher-Karrer, H. Zangl, and M. Hofbaur, "Radar based target tracking and classification for efficient robot speed control in fenceless environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 799–806.

[5] D. Barnes and I. Posner, "Under the radar: Learning to predict robust keypoints for odometry estimation and metric localisation in radar," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Paris, 2020. [Online]. Available: https://arxiv.org/abs/2001.10789

[6] S. May, D. Droeschel, S. Fuchs, D. Holz, and A. Nüchter, "Robust 3d-mapping with time-of-flight cameras," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1673–1678.

[7] J. Steinbaeck, C. Steger, E. Brenner, G. Holweg, and N. Druml, "Occupancy grid fusion of low-level radar and time-of-flight sensor data," in *2019 22nd Euromicro Conference on Digital System Design (DSD)*, 2019, pp. 200–205.

[8] Y. Cheng, J. Su, M. Jiang, and Y. Liu, "A novel radar point cloud generation method for robot environment perception," *IEEE Transactions on Robotics*, pp. 1–20, 2022.

[9] J. Steinbaeck, C. Steger, G. Holweg, and N. Druml, "Design of a low-level radar and time-of-flight sensor fusion framework," in *2018 21st Euromicro Conference on Digital System Design (DSD)*, 2018, pp. 268–275.

[10] F. Zoghlami, O. K. Sen, H. Heinrich, G. Schneider, E. Ercelik, A. Knoll, and T. Villmann, "Tof/radar early feature-based fusion system for human detection and tracking," in *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, vol. 1, 2021, pp. 942–949.

[11] A. P. Sligar, "Machine learning-based radar perception for autonomous vehicles using full physics simulation," *IEEE Access*, vol. 8, pp. 51 470–51 476, 2020.

[12] C. Schöffmann, B. Ubezio, C. Böhm, S. Mühlbacher-Karrer, and H. Zangl, "Virtual radar: Real-time millimeter-wave radar sensor simulation for perception-driven robotics," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4704–4711, 2021.

[13] M. Keller, J. Orthmann, A. Kolb, and V. Peters, "A simulation framework for time-of-flight sensors," in *2007 International Symposium on Signals, Circuits and Systems*, vol. 1, 2007, pp. 1–4.

[14] H. Gietler, B. Ubezio, and H. Zangl, "Tofsim: Real-time tof camera simulation for robotic applications," in *Austrian Robotics Workshop (ARW)*, 2023, to be published.

[15] U. Technologies. (2022) Unity 3d. [Online]. Available: https://Unity3D.com/

[16] B. Walter, S. Marschner, H. Li, and K. Torrance, "Microfacet models for refraction through rough surfaces." *Eurographics Symposium on Rendering*, pp. 195–206, 01 2007.

[17] B. Burley, "Physically-based shading at disney," *ACM SIGGRAPH Course Notes*, pp. 1–7, 2012.

[18] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[19] S. Ortiz, M. A. M. Sathiaseelan, and A. Cha, "Time-of-flight camera characterization with functional modeling for synthetic scene generation," *Opt. Express*, vol. 29, no. 23, pp. 37 661–37 682, Nov 2021.

[20] X. Li, X. Wang, Q. Yang, and S. Fu, "Signal processing for tdm mimo fmcw millimeter-wave radar sensors," *IEEE Access*, vol. 9, pp. 167 959–167 971, 2021.

[21] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 22–35, 2017.

[22] A. Doerry, "Noise and noise figure for radar receivers," in *Sandia National Laboratory Report*, 10 2016.