2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

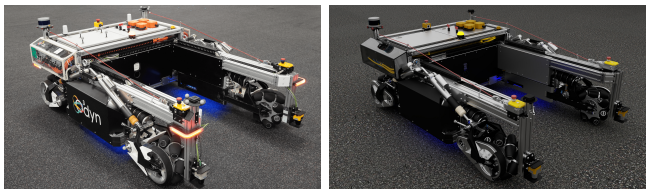
Simulation Modeling of Highly Dynamic Omnidirectional Mobile Robots Based on Real-World Data

Marvin Wiedemann[1], Ossama Ahmed[1], Anna Dieckhöfer[1], Renato Gasoto[2], and Sören Kerner[1,3]

***Abstract*— Simulation is a key technology in robotics as it enables the generation of environmental data and testing scenarios for development and maintenance purposes. However, simulations are an imperfect representation of the real-world and the so-called sim-to-real gap between simulation and reality hinders the deployment of virtual developed solutions without additional effort. Modeling complex systems like highly dynamic and holonomic mobile robots presents additional complexities in simulation. This paper addresses these challenges through a case study on creating a model for a highly dynamic logistics robot. The study considers the modeling of the entire system down to creating suitable colliders for the rollers of a Mecanum wheel. Additionally, the impact of significant physics parameters is presented. To bridge the sim-to-real gap, a pipeline is developed that utilizes a Motion Capture system to compare the behavior of a real robot with its simulated counterpart across various motions. By leveraging expert knowledge gained from the real-world data, the simulation model is manually tuned to replicate complex system behaviors, such as sliding effects.**

I. Introduction

Simulation has an outstanding potential to drive innovations in robotics by accelerating development cycles and providing virtual test environments [1]. Even so simulation models are abstractions of real systems and therefore by definition imperfect. Modeling complex robots [1] and bridging the gap between simulation and reality, known as the sim-to-real gap [2], [3], are the main challenges within the usage of simulation. A common example of a challenging modeling task is the implementation of holonomic robots, whose motion relies on the interaction between the ground and the rollers of an omnidirectional wheel [4]–[6]. These robots are used in various use cases, like mobile manipulators, as their high degree of freedom and agility makes them very efficient [7]–[10]. Simulating omnidirectional robots has a long history, but in most cases, the real-world dynamics are only roughly modeled due to limitations of the simulation tools [11]–[13]. Other work focuses less on the model itself and more on the application, such as localization and navigation [14]. A series of publications [15]–[18] has developed more sophisticated simulation models, but focus only on less dynamic motion. With the increasing number of simulation tools for robotics [19], [20], simulating complex and highly dynamic robots should be feasible as well. However, a survey among developers revealed that the complexity of creating models and the reality gap are the primary reasons for not utilizing simulation [21]. This paper aims to address these challenges by providing guidance on simulating highly dynamic omnidirectional mobile robots and outlining how different methods affect the gap between the resulting model and the real robot. Instead of focusing on mathematical considerations, this paper presents a modeling guideline that highlights key parameters since common simulation tools [22] use a general physics engine to perform rigid body simulation and allow the user to modify the model behavior by changing its attributes such as joint attributes (damping, stiffness, etc.), rigid body inertia properties and collision meshes. This list can grow extensively, so it becomes overwhelming to understand which parameters have the most impact, both in quality of the results or in computational costs. In this paper, NVIDIA Isaac Sim [23] is used, as it fulfills the demand of robotics developers seeking a single simulation tool for various tasks [21]. Isaac Sim combines GPU-based high-quality renderer with the state-of-the-art physics engine PhysX, which is a general purpose whole world physics model based on Extended Positon-based Dynamics constrained optimization. It supports integration with the Robot Operating System (ROS), serves as a training environment for Reinforcement Learning [24], and facilitates the generation of synthetic data, among other features. The modeling approaches are demonstrated using O^{3}dyn [25] (Fig. 1(a)), a logistics robot for autonomous pallet transport indoors and outdoors. O^{3}dyn can move omnidirectionally at velocities up to $10\,\mathrm{m/s}$ and has an air suspension for lifting pallets and driving on uneven terrain. These characteristics make O^{3}dyn an excellent example of a complex robot with high dynamics. To ensure experiment reproducibility, the model (Fig. 1(b)) has been released as open-source [26].



(a) Reality (b) Simulation

Fig. 1: The logistics robot O^{3}dyn.

The contribution of this paper is to establish a process that serves as a guideline for building simulation models using realistic parameters, understand which parameters are the

This research has partly been funded by the Federal Ministry of Education and Research of Germany and the state of North-Rhine Westphalia as part of the Lamarr Institute for Machine Learning and Artificial Intelligence.

[1]M. Wiedemann, O. Ahmed, A. Dieckhöfer, and S. Kerner are with the Department AI and Autonomous Systems, Fraunhofer Institute for Material Flow and Logistics, Dortmund, Germany.

[2] R. Gasoto is with NVIDIA, Santa Clara, CA, USA.

[3] S. Kerner is PI at the Lamarr Institute for Machine Learning and Artificial Intelligence, Dortmund, Germany.

most impactful on the model performance and accuracy, and validate it with the real-world behavior of highly dynamic omnidirectional mobile robots. The process consists of three parts: First, the modeling of omnidirectional dynamics is demonstrated by focusing on the Mecanum wheels and their rollers, providing insights into the effect of different collider models on the driving behavior and computational resources. Second, key physics parameters and their impact on the overall simulation model behavior are outlined through small experiments. Third, real-world data are used to analyze the behavior of the robot and compare simulation and reality. The key parameters are then used to manually optimize the simulation model and reduce the sim-to-real gap.

II. Modeling of Omnidirectional Mobile Robots

Building a robot's simulation model typically starts with importing its construction data from the Unified Robot Description Format (URDF) [27] or Computer Aided Design (CAD) file [28]. For the O^{3}dyn simulation model, shown in Fig. 1(b), the CAD data are imported via Onshape [29]. Afterward, multiple manual steps are performed to replicate the real robot. Since the construction data of O^{3}dyn are used, redundant components, such as screws or wires, are first removed to minimize the required computing resources. Within this step, some textures for the visuals are optimized to represent the real-world appearance. The joints of the robot are automatically generated during import, but some characteristics, such as damping and stiffness for the suspension, wheel drives, and actuators, need to be tuned. To avoid modeling the air suspension mechanism, the dynamics of the suspension are transferred to equivalent joints that replicate its behavior. Then topological loops are marked to maintain a tree structure in the hierarchy. In this paper, the joints of the damping system are chosen to be relatively stiff compared to the real system. This reduces the influence of an O^{3}dyn specific component, and results can be better transferred to other omnidirectional robots. Optionally, sensors such as laser scanners and cameras [30], can be modeled. The simulation model should be part of the ROS ecosystem like the real O^{3}dyn. Therefore, the ROS interfaces are replicated via OmniGraph [31]. Further, the link structure of the model is modified to match the robot's TF-Tree. This step is usually not needed for models imported from URDF. The resulting simulation model, along with a definition of its coordinate system, is shown in Fig. 2(a) and is open-source accessible [26]. For the entire vehicle motion the modeling of the Mecanum wheels is important, each wheel is controlled independently to enable the robot to drive in three directions (longitudinal, lateral, and rotation) [10]. This is modeled by using active revolute joints in simulation. Further, any wheel shown in Fig. 2(b) consists of 7 rollers. These are sized and arranged in a fashion, that a contact between a roller and the ground is guaranteed while the wheel rotates. Gaps between the rollers would lead to small bumps [10]. Therefore, the simulation model of the wheel needs to be perfectly round [9], as shown in Fig. 2(c). The mathematics for controlling the wheels within a holonomic base controller are not described here, the corresponding formulae can be found in one of the numerous sources on this topic [4], [9], [18]. Based on this, the base controller can be implemented either via OmniGraph as well as with a custom node based on the Isaac Sim Python API or ROS2 joints.

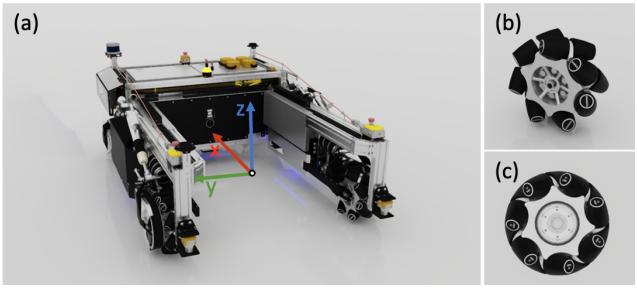


Fig. 2: The simulation model of O^{3}dyn (a) with coordinate system and its Mecanum wheel in two perspectives (b), (c).

A crucial aspect of the wheel modeling is the roller collider. Using a high-poly mesh to accurately represent the original wheel seems a good choice, but it leads to many contact points during motion and therefore to high computational costs. To avoid a reduced real-time performance, [17] and [18] use low-poly models, while [4] even models a custom mesh consisting of 5 spheres. In this paper the impact of nine different collider models on the robot's movement is analyzed, a selection of the types is shown in Fig. 3. Most meshes are manually created and aim to accurately align the geometric bodies of the CAD data of the rollers. The first type uses cylinders, which intuitively match the shape of the rollers (4C, 7C). The second type consists of a varying number of spheres (4S, 6S, 8S, 10S), but excludes a sphere in the middle due to the presence of a notch for the roller bearing. However, an optimized version (O) with 11 spheres is developed, which includes a sphere in the middle to provide smoother ground contact. Additionally, two auto-generated colliders based on CAD data are being analyzed: Convex Hull (CH) and Convex Decomposition (CD).

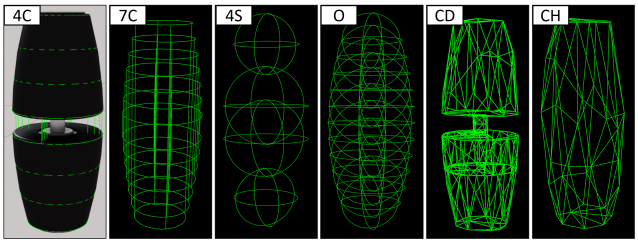


Fig. 3: Selection of the collider models used to represent the rollers of the Mecanum wheel.

The impact of the models is analyzed in three experiments. O^{3}dyn is driven along a straight line with a slow velocity of v = $0.1\,\mathrm{m/s}$ using an open-loop controller, as the focus lies on the contact between the collider and the ground. The simulation model is commanded to drive for $4\,\mathrm{m}$ along the longitudinal and lateral directions and the resulting drift is measured. The results, given in Table I, indicate that CH and CD perform worst in terms of keeping the straight motion.

When comparing this auto-generated meshes (CH and CD) with the colliders shown in Fig. 3, there seems to be a correlation between the smoothness of the approximations and the drift performance. Moreover, spherical models seem to exhibit the least drift along the lateral direction, while both spherical and cylindrical approximations have comparable drift performance along the longitudinal direction.

TABLE I: Drift performance of simulated O^{3}dyn for different collider models of the rollers in mm.

Longitudinal Motion

Model	C4	C7	S4	S6	S8	S10	O	CD	CH
$\lvert\Delta\rvert$ **y**	3.0	1.42	2.9	2.1	1.7	1.5	1.48	33	55

Lateral Motion

Model	C4	C7	S4	S6	S8	S10	O	CD	CH
$\lvert\Delta\rvert$ **x**	4.0	8.0	2.3	2.6	4.7	6.0	5.0	40	116

To investigate the cause of the drift, the motion of the center of the wheel along the Z-axis is analyzed while performing a longitudinal motion at a speed of v = $0.1\,\mathrm{m/s}$. Within Fig. 4 the data for the colliders with the smallest and largest drift are compared. The comparison illustrates the periodic motion resulting from the contact between roller and ground during wheel rotation. One aspect to consider is the smoothness of the motion, which can be analyzed by observing the amplitude and the bumps in one cycle. The optimized model (O) exhibits a smoother motion then the CH approach. This can be correlated to the drift performance, shown in Table I.

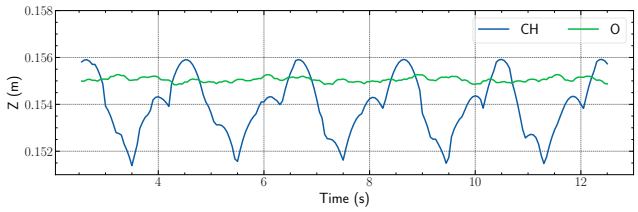


Fig. 4: The motion of the center of the wheel of O^{3}dyn for the O and CH collider models of the rollers.

Using a high-poly mesh as a collider model improves driving smoothness but also increases computational cost. To compare the computational effort of different models, O^{3}dyn is simulated for $100\,\mathrm{s}$ in real-time, while the corresponding simulation time is measured. Similar to [32], a real-time factor is calculated as the ratio of the simulated time to the time elapsed in reality. It is important to note that the calculated values depend on the resources of the test machine and other factors, making them non-reproducible. However, the values shown in Table II are intended to compare the different models. The trend between the models is reproducible and higher values indicate faster running simulations and lower computational costs. The values contain a clear pattern that the computational time depends on the number of contact points. Increasing the number of geometric objects used for modeling leads to a decrease in real-time performance, as indicated by smaller values. Additionally, using spheres as colliders is less computationally intensive than cylinders for the same number of geometric objects. Spherical objects require fewer iterations in the collision detection broad phase during the physics run in the simulation [33].

TABLE II: Real-time factor for different collider models.

C4	**C7**	**S4**	**S6**	**S8**	**S10**	**O**	**CD**	**CH**
0.70	0.60	0.83	0.76	0.69	0.65	0.62	0.76	0.82

Overall, the experiments highlight the significant impact of collider modeling on the motion of the robot. It is crucial to design the colliders smoothly to minimize jumps in the Z-direction caused by collider spikes interacting with the ground. Manual modeling of the rollers is recommended, and spheres prove to be a suitable choice. Based on the results shown in Table I and Table II, the S6 model has demonstrated an effective balance between the drift performance and the computational effort. As a result, it has been selected to model the roller in the experiments in the following sections.

III. Physics Parameters

By incorporating suitable wheel colliders, a simulation model can accurately replicate the omnidirectional dynamics, but physical effects, such as sliding, have a significant impact. To replicate such behaviors, key physics parameters within the simulation are investigated.

A. Dynamic Friction

Dynamic friction has a significant role in governing the motion of omnidirectional robots. It mainly arises because of the contact between the rollers and the underlying surface, causing resistance that can impact speed, control, and overall performance [34]. To showcase the impact of the parameter, the speed trajectory of the simulated O^{3}dyn is investigated for different friction coefficients when applying a step velocity command of $5\,\mathrm{m/s}$. The resulting trajectories in Fig. 5 show that with a smaller friction coefficient, more time is needed so that O^{3}dyn can reach the target velocity. Although the measured wheel speed is constant for all tests, the embedded PD controller of the joints [33] must spend more time for acceleration. Such behavior can be explained by the fact that for the runs with smaller dynamic friction, the wheels tend to have a larger slip ratio. Therefore, they require more time to be able to pick up the speed as less traction force is generated from the contact between the rollers and the ground surface. This effect is further highlighted by the zero dynamic friction coefficient case in which the model is not capable of accumulating any speed.

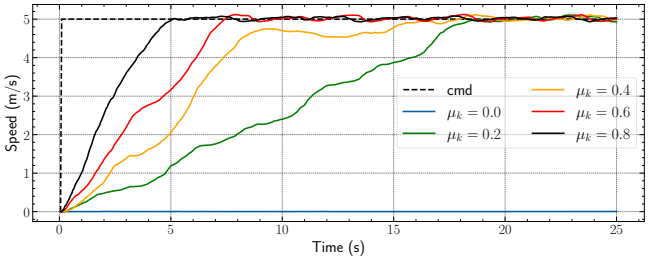


Fig. 5: Response of O^{3}dyn for a step speed command with different dynamic friction coefficients μ_k.

B. Center of Mass

The center of mass (CoM), often referred to as the balance point of the robot, defines the distribution of the weight of the robot. Variations in the CoM position have a profound effect on the omnidirectional robot performance. Here, the effect of shifting the CoM on the longitudinal and the lateral motion is investigated. The simulation model is commanded to drive along a straight line in both the longitudinal and lateral direction with a speed of $v_x = 2\,\mathrm{m/s}$ and an acceleration of $a_x = 2\,\mathrm{m/s^2}$. For the longitudinal motion, the CoM is shifted along the Y-axis of O^{3}dyn with a magnitude of $0.2\,\mathrm{m}$ and $0.5\,\mathrm{m}$ in each direction. The resulting trajectory for the longitudinal motion in Fig. 6 shows that the more the CoM is shifted in one direction along the Y-axis, the more drift it causes in the opposite direction.

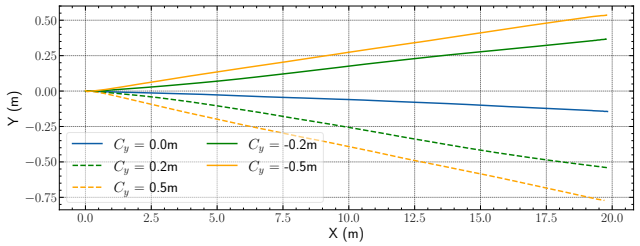


Fig. 6: Drift along the Y-axis caused by shifting the CoM of the simulated O^{3}dyn while undergoing a longitudinal motion.

A similar experiment is performed for the lateral motion. The CoM is shifted along the X-axis of O^{3}dyn with a magnitude of $0.1\,\mathrm{m}$ and $0.2\,\mathrm{m}$ in both directions. The resulting motion, shown in Fig. 7, underlines the same trend observed in Fig. 6. However, one clear difference between the two motions is the sensitivity to the change in the CoM position. While in the case of performing a longitudinal motion, shifting the CoM $0.5\,\mathrm{m}$ along the Y-axis only results in $0.8\,\mathrm{m}$ drift, shifting the CoM $0.1\,\mathrm{m}$ along X-axis during lateral motion results in a drift of nearly $5.0\,\mathrm{m}$.

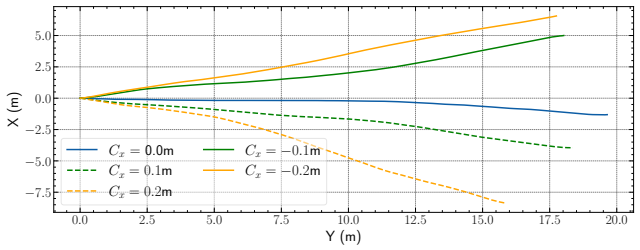


Fig. 7: Drift along the X-axis caused by shifting the CoM of the simulated O^{3}dyn while undergoing a lateral motion.

IV. Reducing the Sim-to-Real Gap

The insights on the physics parameters can be utilized to modify the model behavior of O^{3}dyn to reduce the sim-to-real gap. To gain insights on the real robot, its position is tracked with a Motion Capture (MoCap) system while driving various trajectories, similar to [35]. A pipeline, shown in Fig. 8, is used to execute the maneuvers in reality and simulation. It is built based on ROS2 and generates trajectories consisting of three phases: acceleration, constant speed, and deceleration. They are specified by the acceleration (a_x, a_y), the maximum velocity (v_x, v_y) and the length (X_{dist}, Y_{dist}). The target velocities are communicated via the topic /cmd_vel with $f = 100\,\mathrm{Hz}$ and stored as a bag, a ROS file format used to record and replay topics. In this way, the real and the simulated O^{3}dyn can be driven reproducibly. A Message Timing node is used to time the messages of the bag files to guarantee the sync with simulation time. The simulation model delivers its ground truth data via TF messages, while the real robot is tracked with a Vicon MoCap system. The position data are stored in bag files and can be used to determine the sim-to-real gap via visual inspection or by calculating the error between the pose data.

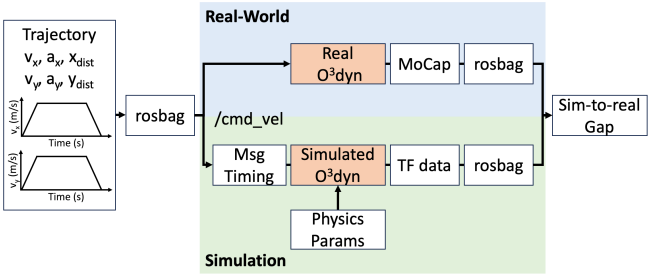


Fig. 8: The architecture of the pipeline used to command customized trajectories to compare simulation and reality.

To gain a comprehensive understanding of O^{3}dyn's system behavior, different longitudinal and lateral trajectories are driven covering a wide range of straight-line dynamics. To ensure safety, the maximum velocity of O^{3}dyn and the length of the traveled distance are limited to align with the size of the research hall. Although, with a speed of $v_x = 4\,\mathrm{m/s}$ and O^{3}dyn's maximum acceleration of $a_x = 3.85\,\mathrm{m/s^2}$, the investigated dynamics are high in comparison to other works [16], [17]. The lateral motion is limited to $v_y = 3\,\mathrm{m/s}$, $a_y = 3\,\mathrm{m/s^2}$ as the robot exhibits reduced system control in this case. All experiments are performed with an open-loop controller, and each trajectory is repeated ten times to account for the variance of the runs. Since a consistent system behavior can be observed within the experiments, the following only includes the data for the least and most dynamic cases for longitudinal and lateral motion.

A. Longitudinal Motion

In all experiments with longitudinal motion, the real O^{3}dyn deviates to the right. Since the effect increases with higher dynamics, the most dynamic trajectory is investigated. The pose data in Fig. 9 show a deviation of around -1.5 meter for Y at the end. Based on the Yaw data, it can be inferred that O^{3}dyn rotates in the negative Yaw direction during the acceleration phase. This modified orientation causes a drift in the negative Y-direction while driving at a constant speed. During deceleration, O^{3}dyn rotates in the opposite direction, but its impact on the Y-deviation is small since the driving is not continued. The trajectory is repeated in simulation with a reference parameter set (RP) (no CoM shift, μ_k = 0.7). The corresponding graphs in Fig. 9 outline no differences between the simulated and real pose data in

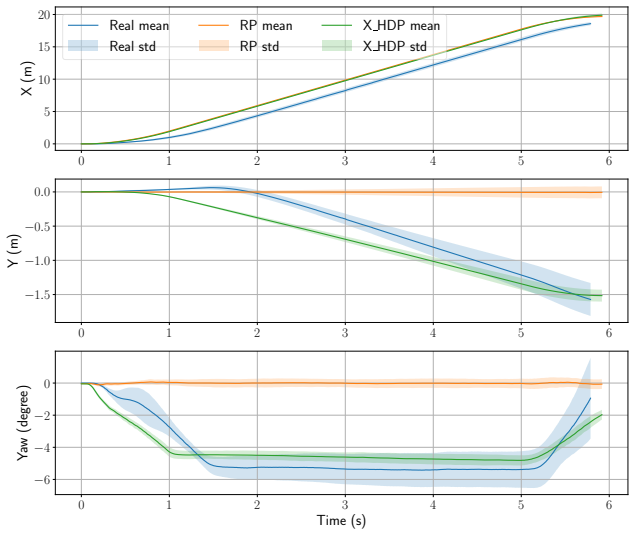


Fig. 9: Sim-to-real comparison for longitudinal motion for high dynamics (v_x = 4 m/s, a_x = 3.85 m/s^2, x_{dist} = 20 m).

X but considerable differences in Y and Yaw. The model drives a straight line and does not replicate the real-world sliding, so no drift happens. To reduce the sim-to-real gap, the insights from Section III are utilized to modify the model behavior. According to the CoM experiment (Fig. 6), a shift of the CoM in positive Y-direction is needed to replicate the negative Y-drift of the real robot. Since this only causes a lateral drift and fails to replicate the rotation, a shift of the CoM in X-direction is also needed. Considering the u-shaped construction form of O^{3}dyn, a shift to the front is reasonable. Afterward, the simulated O^{3}dyn shows a rotation behavior, and the experiment is repeated multiple times while tuning the parameters. The friction coefficient is used to modify the magnitude of the rotation and the variety between different runs. To simplify the process of finding suitable simulation parameters, the CoM is not modified in Z-direction. A Z-shift could also cause a change in the X-Y-data due to the damping behavior of O^{3}dyn, but the influence is less significant and is therefore neglected here to simplify tuning. The finally tuned parameter set (X_HDP), as well as the sets for all the following experiments, are listed in Table III.

TABLE III: The used physics parameter sets tuned to replicate different dynamics in simulation.

	RP	**X_LDP**	**X_HDP**	**Y_LDP**	**Y_HDP**
C_x (m)	0.0	0.0	0.2	0.022	0.08
C_y (m)	0.0	0.69	0.45	-0.1	0.2
μ_k	0.70	0.75	0.75	0.4	0.45

The resulting graph for the tuned parameters (X_HDP) in Fig. 9 demonstrates that the simulation model can replicate the behavior of the real robot at high dynamics. The difference between the graphs in the acceleration phase can be explained by unintended acceleration dips caused by the controller of the real systems, which were observed during the measurements. The simulation model properly simulates the complex rotation behavior resulting from the sliding effects of the real system. In contrast, significant differences are evident when comparing the graphs of the simulation model with the same parameter set X_HDP and the real robot at lower velocities, as shown in Fig. 10. The corresponding graph shows a slow rotation to the left (+Yaw) for the simulated O^{3}dyn, leading to a drift in the positive Y-direction. Still, the real O^{3}dyn drifts in the negative Y-direction due to a small rotation to the right (-Yaw). Since the lower dynamics require a stronger rotation during acceleration to counteract the ongoing rotation to the left, another parameter set, X_LDP, was explicitly tuned for low dynamics. This proves to be more challenging. The final values for X_LDP in Table III are notable for their significant shift of the CoM, which ensures that the initial rotation dominates the final deviation. Using this new parameter set, the simulation better replicates the real-world X- and Y-pose in Fig. 10.

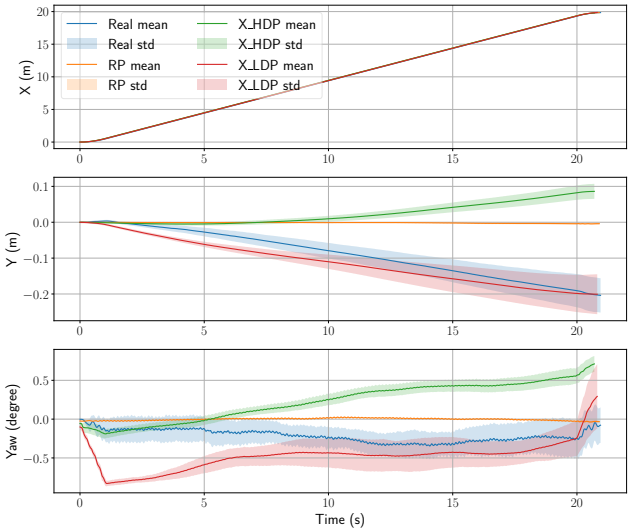


Fig. 10: Sim-to-real comparison for longitudinal motion for low dynamics (v_x = 1 m/s, a_x = 1 m/s^2, x_{dist} = 20 m).

B. Lateral Motion

Within the experiments on lateral motion, a rotation over the whole drive can be clearly observed for the real robot, with fewer discrete sliding events for acceleration or deceleration in comparison to the longitudinal motion. For high dynamics, the driven path even appears to be more of a curve than a straight line. This can be seen in the graphs in Fig. 11 and Fig. 12, which show the pose data using the Y-position on the X-axis instead of the time to focus on the shape of the trajectories. To replicate the real-world behavior, again parameter sets are tuned for small dynamics (Y_LDP) and high dynamics (Y_HDP) given in Tab. III. During the tuning, some strange jumps and inconsistencies within the lateral motion are observed in simulation. This indicates that the simulation needs more computational time to calculate the contact forces between the rollers and the ground. Therefore, the parameter *Time Steps per Second* needs to be considered. It defines how many steps the simulation uses to simulate one second – the more collisions have to be calculated, the more time steps are necessary. To be able to simulate

the lateral motion properly, the parameter value is doubled from 360 to 720. Referring to the tuned parameter sets in Tab. III, the key value for tuning is the shift of the CoM in the X-direction, which is extremely sensitive. Since opposite wheels rotate in the opposite direction for lateral movement, higher contact forces on the front or rear wheels cause a rotation component because of stronger contact forces. Due to the U-shape of O^{3}dyn, the CoM is located at the front part. This causes the curved trajectories for the real robot and with this knowledge, the behavior can be replicated in simulation as shown in Fig. 11 and Fig. 12. Another observation is the traveled distance, the real-world O^{3}dyn does not reach the commanded $10\,\mathrm{m}$ or $20\,\mathrm{m}$. This indicates that the rotation of the wheels is not perfectly translated in Y-movement because of slip. As O^{3}dyn is controlled using an open-loop controller, this effect remains uncompensated, resulting in the commanded velocity not being reached. To replicate this behavior in simulation, the dynamic friction value is reduced significantly. In the end, the behavior of the simulation model is much closer to reality, but there is still a gap between the length of the real and simulated trajectories in Fig. 12. Improving the model further based on the used physics parameters is very difficult, since small changes have a huge impact on the lateral motion. This is underlined by the graph of Y_HDP in Fig. 11. Its huge standard deviation shows that poorly chosen physics parameters can result in an extremely varying simulation behavior. Further the trend of the simulation is totally different to the real O^{3}dyn.

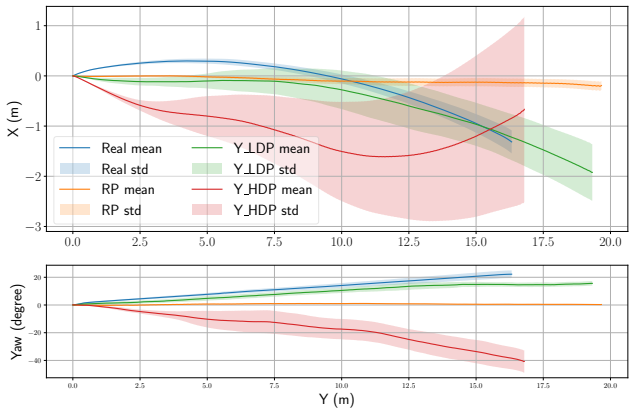


Fig. 11: Sim-to-real comparison for lateral motion with low dynamics ($\mathrm{v}_y = 1\,\mathrm{m/s}$, $\mathrm{a}_y = 1\,\mathrm{m/s^2}$, $\mathrm{y}_{dist} = 20\,\mathrm{m}$).

As depicted in Fig. 12, the real-world trajectory for highly dynamic lateral motion exhibits a curved shape and is considerably shorter in the Y-direction. The tuned physics parameters in Y_HDP enable the model to replicate this trend, resulting in a matching end position in simulation and reality. This emphasizes that the reproduction of the real-world behavior for different dynamics using physics parameters is achievable, but it fails to find a single set of parameters that accurately mimics a diverse range of dynamics. Further, selecting suitable physics parameters for lateral motion is difficult, since the model is sensitive to small changes and requires small friction coefficients.

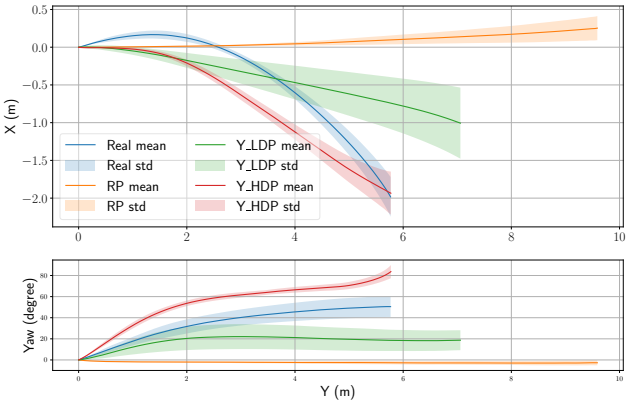


Fig. 12: Sim-to-real comparison for lateral motion with high dynamics ($\mathrm{v}_y = 3\,\mathrm{m/s}$, $\mathrm{a}_y = 3\,\mathrm{m/s^2}$, $\mathrm{y}_{dist} = 10\,\mathrm{m}$).

V. Conclusion

This paper presents a guideline for modeling omnidirectional mobile robots using the logistics robot O^{3}dyn and identifying key simulation parameters to tune when matching simulation to reality. First, the impact of collider modeling for the Mecanum rollers on the driving behavior and computational needs was analyzed. Manual collider models outperformed auto-generated meshes, and the use of spheres provided a good performance-to-cost balance. Second, the effect of key physics parameters was demonstrated. Although the list of parameters presented is not exhaustive, significant changes in the model's behavior were achieved. Third, a pipeline for comparing real and simulated robots was introduced and used to analyze the real robot. This knowledge was then utilized to manually tune the simulation model, reducing the sim-to-real gap. The results showed the ability to model complex system behaviors, including slipping, for both low and high dynamics. This opens opportunities for developing algorithms within simulation, such as a closed-loop controller to compensate the discrepancy between commanded and driven lateral velocities. A separate set of parameters was required for each trajectory, indicating discrepancies between the real and simulated O^{3}dyn. While these discrepancies can be overcompensated through manual tuning, they still result in a significant sim-to-real gap when considering all dynamics. Improving the mass distribution and damping system in the simulation model could help identify a more generic parameter set. Further investigation is needed to determine the applicability of the presented methods on simpler robots. Dynamic adaptation of physics parameters may provide an alternative solution. Moreover, the experiments on various dynamics revealed the challenges associated with manual tuning of the parameters. Future work could extend the pipeline by integrating system identification approaches to automatically learn the system behavior from real-world data.

Acknowledgment

We would like to thank Jan Finke and Mrunal Hatwar for their support within the sim-to-real experiments, as well as Steffen Daniel for his work on modeling the colliders. Further, we are grateful for the support of the O^{3}dyn team.

REFERENCES

[1] H. Choi, C. Crump, C. Duriez, A. Elmquist, G. Hager, D. Han, F. Hearl, J. Hodgins, A. Jain, F. Leve, C. Li, F. Meier, D. Negrut, L. Righetti, A. Rodriguez, J. Tan, and J. Trinkle, "On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward," *Proceedings of the National Academy of Sciences*, vol. 118, Jan. 2021.

[2] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *Advances in Artificial Life*, ser. Lecture Notes in Computer Science, F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, Eds. Berlin, Heidelberg: Springer, 1995, pp. 704–720.

[3] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, pp. 737–744.

[4] G. Bayar and S. Ozturk, "Investigation of the effects of contact forces acting on rollers of a mecanum wheeled robot," *Mechatronics*, vol. 72, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957415820301318

[5] T. Giurgiu, G. Bârsan, I. Virca, and C. Pupăză, "Mecanum wheeled platforms for special applications," in *International conference KNOWLEDGE-BASED ORGANIZATION*, vol. 28, no. 3, 2022, pp. 44–51.

[6] G. Indiveri, "Swedish wheeled omnidirectional mobile robots: Kinematics analysis and control," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 164–171, 2009.

[7] K. Kanjanawanishkul, "Omnidirectional wheeled mobile robots: Wheel types and practical applications," *International Journal of Advanced Mechatronic Systems*, vol. 6, p. 289, Feb. 2015.

[8] T. Giurgiu, G. Bârsan, I. Virca, and C. Pupăză, "Mecanum Wheeled Platforms for Special Applications," *International conference KNOWLEDGE-BASED ORGANIZATION*, vol. 28, no. 3, pp. 44–51, Jun. 2022. [Online]. Available: https://www.sciendo.com/article/10.2478/kbo-2022-0086

[9] N. Tlale and M. de Villiers, "Kinematics and Dynamics Modelling of a Mecanum Wheeled Mobile Platform," in *2008 15th International Conference on Mechatronics and Machine Vision in Practice*, Dec. 2008, pp. 657–662.

[10] S. Dickerson and B. Lapin, "Control of an omni-directional robotic vehicle with Mecanum wheels," in *NTC '91 - National Telesystems Conference Proceedings*, Mar. 1991, pp. 323–328.

[11] A. Koestler and T. Braunl, "Mobile robot simulation with realistic error models," *International Conference on Autonomous Robots and Agents, ICARA 2004*, pp. 46–51, 01 2004.

[12] O. Michel, "Cyberbotics Ltd. Webots™: Professional Mobile Robot Simulation," *International Journal of Advanced Robotic Systems*, vol. 1, no. 1, p. 5, Mar. 2004, publisher: SAGE Publications. [Online]. Available: https://doi.org/10.5772/5618

[13] J. Goncalves, J. Lima, H. Oliveira, and P. Costa, "Sensor and actuator modeling of a realistic wheeled mobile robot simulator," in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*. Hamburg, Germany: IEEE, Sep. 2008, pp. 980–985. [Online]. Available: http://ieeexplore.ieee.org/document/4638513/

[14] H. Quang, T. Manh, C. Manh, P. Tin, M. Van, N. Tien Kiem, and D. Nguyen Duc, "An Approach to Design Navigation System for Omnidirectional Mobile Robot Based on ROS," *International Journal of Mechanical Engineering and Robotics Research*, pp. 1502–1508, Jan. 2020.

[15] Y. Li, S. Dai, Y. Zheng, F. Tian, and X. Yan, "Modeling and Kinematics Simulation of a Mecanum Wheel Platform in RecurDyn," *Journal of Robotics*, vol. 2018, Jan. 2018, publisher: Hindawi. [Online]. Available: https://www.hindawi.com/journals/jr/2018/9373580/

[16] Y. Li, S. Ge, S. Dai, L. Zhao, X. Yan, Y. Zheng, and Y. Shi, "Kinematic Modeling of a Combined System of Multiple Mecanum-Wheeled Robots with Velocity Compensation," *Sensors*, vol. 20, no. 1, p. 75, Jan. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/1/75

[17] Y. Li, S. Dai, Y. Shi, L. Zhao, and M. Ding, "Navigation Simulation of a Mecanum Wheel Mobile Robot Based on an Improved A* Algorithm in Unity3D," *Sensors*, vol. 19, no. 13, p. 2976, Jan. 2019, number: 13 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1424-8220/19/13/2976

[18] A. Apurin, B. Abbyasov, A. Dobrokvashina, Y. Bai, M. Svinin, and E. Magid, "Omniwheel Chassis' Model and Plugin for Gazebo Simulator," *Proceedings of International Conference on Artificial Life and Robotics*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:259886517

[19] J. Collins, D. Howard, and J. Leitner, "Quantifying the reality gap in robotic manipulation tasks," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6706–6712.

[20] P. Castillo-Pizarro, T. V. Arredondo, and M. Torres-Torriti, "Introductory Survey to Open-Source Mobile Robot Simulation Software," in *2010 Latin American Robotics Symposium and Intelligent Robotics Meeting*, Oct. 2010, pp. 150–155.

[21] A. Afzal, D. S. Katz, C. Le Goues, and C. S. Timperley, "Simulation for robotics test automation: Developer perspectives," in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, 2021, pp. 263–274.

[22] J. Collins, S. Chand, A. Vanderkop, and D. Howard, "A Review of Physics Simulators for Robotic Applications," *IEEE Access*, vol. 9, pp. 51 416–51 431, 2021.

[23] "Nvidia isaac sim," https://developer.nvidia.com/isaac-sim, accessed: 2023-09-11. [Online]. Available: https://developer.nvidia.com/isaac-sim

[24] J. Eßer, N. Bach, C. Jestel, O. Urbann, and S. Kerner, "Guided reinforcement learning: A review and evaluation for efficient and effective real-world robotics [survey]," *IEEE Robotics Automation Magazine*, vol. 30, no. 2, pp. 67–85, 2023.

[25] N. Ullrich and M. Gössner, "Analyse und Optimierung eines FTF-Fahrwerks mit Mecanum-Rädern," *Volume 2022*, p. Issue 18, 2022, medium: application/pdf Publisher: Wissenschaftliche Gesellschaft für Technische Logistik. [Online]. Available: https://www.logistics-journal.de/proceedings/2022/5606

[26] "O3dyn simulation model," git.openlogisticsfoundation.org, Open Logistics Foundation, accessed: 2023-08-09. [Online]. Available: https://git.openlogisticsfoundation.org/silicon-economy/simulation-model/o3dynsimmodel

[27] "Universal robotic description format," wiki.ros.org, accessed: 2023-09-11. [Online]. Available: http://wiki.ros.org/urdf/XML/model

[28] "Cad converter," docs.omniverse.nvidia.com, accessed: 2023-09-11. [Online]. Available: https://docs.omniverse.nvidia.com/extensions/latest/ext_cad-converter.html

[29] "Onshape," onshape.com, accessed: 2023-09-13. [Online]. Available: https://www.onshape.com/de/

[30] "Nvidia isaac sim ros interface," docs.omniverse.nvidia.com, accessed: 2023-09-11. [Online]. Available: https://docs.omniverse.nvidia.com/isaacsim/latest/manual_isaac_extensions.html#sensor-extensions

[31] "OmniGraph," docs.omniverse.nvidia.com, accessed: 2023-08-15. [Online]. Available: https://docs.omniverse.nvidia.com/isaacsim/latest/tutorial_gui_omnigraph.html

[32] Y. Okada, K. Oguma, K. Gunji, Y. Yokota, H. Aryadi, S. Kojima, R. Bezerra, M. Konyo, K. Ohno, and S. Tadokoro, "Fast and accurate simulation of mecanum wheels with passive rollers emulated by fixed joints and anisotropic friction," in *2023 21st International Conference on Advanced Robotics (ICAR)*, 2023, pp. 592–598.

[33] "Nvidia physx documentation," nvidia-omniverse.github.io, accessed: 2023-09-11. [Online]. Available: https://nvidia-omniverse.github.io/PhysX/

[34] C. Ren and S. Ma, "Dynamic modeling and analysis of an omnidirectional mobile robot," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 4860–4865.

[35] A. Kanwischer and O. Urbann, "A machine learning approach to minimization of the sim-to-real gap via precise dynamics modeling of a fast moving robot," pp. 349–354, 2022.