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

STARK: A Unified Framework for Strongly Coupled Simulation of Rigid and Deformable Bodies with Frictional Contact

José Antonio Fernández-Fernández[1] Ralph Lange[2] Stefan Laible[2] Kai O. Arras[2,3] Jan Bender[1]

***Abstract*— The use of simulation in robotics is increasingly widespread for the purpose of testing, synthetic data generation and skill learning. A relevant aspect of simulation for a variety of robot applications is physics-based simulation of robot-object interactions. This involves the challenge of accurately modeling and implementing different mechanical systems such as rigid and deformable bodies as well as their interactions via constraints, contact or friction. Most state-of-the-art physics engines commonly used in robotics either cannot couple deformable and rigid bodies in the same framework, lack important systems such as cloth or shells, have stability issues in complex friction-dominated setups or cannot robustly prevent penetrations. In this paper, we propose a framework for strongly coupled simulation of rigid and deformable bodies with focus on usability, stability, robustness and easy access to state-of-the-art deformation and frictional contact models. Our system uses the Finite Element Method (FEM) to model deformable solids, the Incremental Potential Contact (IPC) approach for frictional contact and a robust second order optimizer to ensure stable and penetration-free solutions to tight tolerances. It is a general purpose framework, not tied to a particular use case such as grasping or learning, it is written in C++ and comes with a Python interface. We demonstrate our system's ability to reproduce complex real-world experiments where a mobile vacuum robot interacts with a towel on different floor types and towel geometries. Our system is able to reproduce 100% of the qualitative outcomes observed in the laboratory environment. The simulation pipeline, named *Stark* (the German word for *strong*, as in strong coupling) is made open-source.**

I. INTRODUCTION

Simulation has become an increasingly important part of robot learning, design, and testing. Examples range from corner case identification for self-driving cars, data generation for robot reinforcement learning, or shorter development and testing cycles for consumer robots. One aspect to this end is accurate simulation of robot-object interaction for which a number of popular physics engines can be used: ODE [2], Bullet [3], MuJoCo [4], Dart [5], Chrono [6], and PhysX [7]. Here, we are interested in the simulation of deformable objects as they are an integral part of everyday life. Robotic bin picking systems in intra-logistics, for example, must be able to handle a variety of flexible, fabric or limp objects and robot vacuum cleaners have to deal with household objects from plants, clothing to carpets. Precise models for contact, friction and material response, together with adequate solvers for systems of deformables *and* rigid bodies are essential to simulate such interactions accurately.

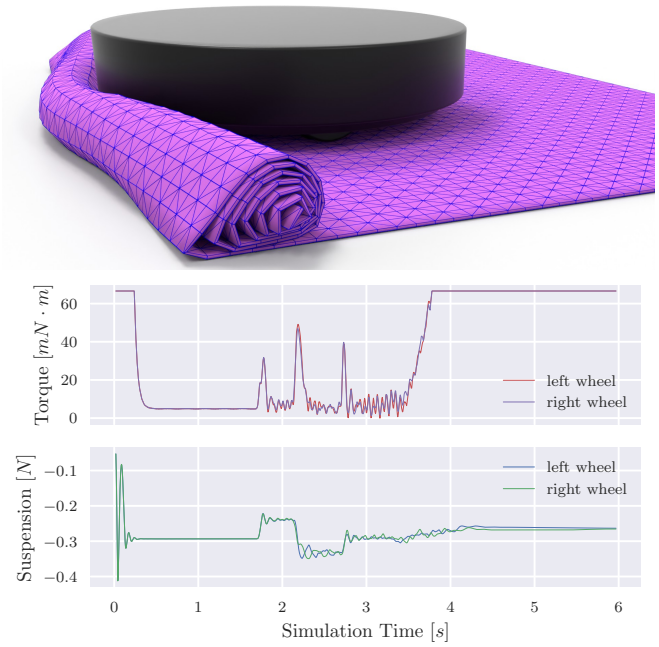

Fig. 1: Simulation of a vacuum cleaning robot (rigid body system) driving straight into a (deformable) towel rolled on the floor. The robot drives over the loose end of the towel and then deforms it but cannot progress, which is the typical behavior obtained in our lab experiments. Important metrics gathered during simulation are shown below: on top, the motor torque, relevant to study the robot power limits and battery consumption, and below, suspension compression, related to the robot's ability to adapt to obstacles.

Often, simulation of different mechanical systems (e.g. cloth, volumetric deformable objects, rigid bodies, etc.) is handled in two stages: the different systems are first considered in isolation, sometimes even with different simulation frameworks, followed by the inter-system interactions, such as the interaction between the wheel and the towel in Fig. 1. Such *weak coupling* can lead to inaccurate results, or even robustness issues, as errors are not minimized in a global sense and might be simply transferred between systems during the coupling iterations. This problematic is magnified in the presence of complex and stiff interactions between the different systems – such as the soft cloth being squeezed between the robot and the floor in Fig. 1. In contrast, *strongly coupled* simulations handle all the systems in a single stage, including the interactions between them. This often results in more

[1]RWTH Aachen University, Computer Animation Group, Germany
✉ lastname@cs.rwth-aachen.de
[2]Bosch Research, Renningen, Germany
✉ firstname.lastname@de.bosch.com
[3]Socially Intelligent Robotics Lab, University of Stuttgart, Germany

stable results in challenging scenarios, but usually comes at a higher computational cost. Second-order methods, which use second-order derivative information, are often preferred in this context as they provide better guarantees than first-order methods, which only use gradient information, such as Position Based Dynamics [8] or Projective Dynamics [9]. Especially when simulating stiff materials or complex contact configurations, the convergence of first-order methods can grind to a halt [10]. Implementation complexity however is typically significantly higher in second-order methods since global data structures, representing all the involved physical systems and interactions, must be built to solve large linear systems of equations. This complexity is further exacerbated in simulations of multiple interacting systems as an increasing number of different types of primitive pairs for collision and friction must be accounted for.

We use a unified global potential energy formulation to model all phenomena in the simulation, from internal mechanical stresses to joints and collisions, for which a minimizer represents a configuration where the global balance of forces is satisfied. Such approach is well-established and can be found in related works such as [11], [12], [13]. Beyond the benefits it provides in terms of robustness, this formulation has the additional advantage that it concentrates the definition of the problem into a single global expression. Our proposed simulation pipeline leverages this to its advantage and uses SymX, a symbolic differentiation tool [14], to automate the generation of first- and second-order derivatives of all energy potentials in the simulation, which otherwise would be a rather time intensive task to do manually. Very importantly, the differentiation pipeline is also made available to users, facilitating expanding on the default capabilities and ensuring that new models will be resolved in a strongly coupled manner together with the existing ones.

In detail, our contributions are:

1) A complete simulation pipeline named *Stark* with support for rigid body systems with joints and non-linear deformable materials (volumetric and cloths/shells), with frictional contacts between all of them. Stark is open-source, operates under a permissive Apache 2.0 license and it is publicly hosted in GitHub. The simulation pipeline uses state-of-the-art second-order optimization time integration techniques, which coupled with the IPC model for contacts, ensures penetration-freeness and great stability even for very challenging simulations. It is shipped with non-linear constitutive models for strain, integrated with FEM, and strain limiting, as well as bending for cloths and shells. The proposed system employs a clean C++ API with Python bindings for programmatically modeling robotic systems and environments and it is designed so that it can be easily expanded with new materials, joints or constraints.
2) We present simulations involving non-linear materials undergoing large deformations coupled with rigid body systems connected with joints and powered by motors. Most notably, we validate Stark's capabilities to reproduce the outcome of different scenarios carried out in a laboratory setting. In particular, we study the interactions between a mobile vacuum cleaning robot and a towel in different configurations and types of floor. In a second experiment we simulate a parallel jaw gripper grasping a deformable plastic cup, and verify that the simulation follows the outcome predicted by static friction analysis.

The paper is structured as follows: we discuss related work in Sec. II, highlight the main components used in our framework in Sec. III, present our validation experiments in Sec. IV and conclude the paper in Sec. V.

II. RELATED WORK

ODE [2], Bullet [3], MuJoCo [4], Dart [5], Chrono [6], and PhysX [7] are amongst the most commonly used physics engines for simulating entire robotic applications. They are also used in robotics simulators like Gazebo [15], Webots [16], CoppeliaSim (formerly V-Rep) [17], and Isaac Sim [18] that provide further visualization, sensor simulations, and interfaces to robotic software frameworks. Liu et al. [1] recently presented a discussion on the importance of physics-based simulators in robotics. Körber et al. [19] and Erez et al. [20] carry our experimental comparisons of these physics engines and simulators, in the context of rigid body simulation.

Simulators with focus on real-time performance (all of the above except Chrono) do not provide the required features to run the complex simulations we aim for. The lack of proper continuum mechanics based material modeling or the very limited self-collision resolution guarantees for deformable objects makes it impossible to handle a simulation like the one shown in Figure 1. While it is possible to couple deformable and rigid bodies simulators [21], such coupling approaches require expert knowledge not only of the mechanics of the problem but also of the simulators themselves, potentially involving significant internal modifications to achieve true strong coupling.

Recently, PhysX has been integrated in a larger proprietary GPU-based physics simulation environment for reinforcement learning research, called IsaacGym, and support for linear continuous materials with FEM is added [22]. Huang et al. [23] demonstrate that the FEM model provided in IsaacGym performs well on a variety of grasping simulations validated with real-world experiments, although it does not provide very accurate results with shell-like objects. Further, it was recently demonstrated that the IPC model for frictional contact outperforms the contact model provided by IsaacGym [24] in a collection of grasping simulations validated with real world experiments. Chrono [6] supports simulation of deformable solids via linear and non-linear FEM for volumetric objects and shells, rigid body systems and frictional contact, which makes it the natural choice for our target simulations from the aforementioned available solutions. However, we were unable to obtain satisfactory results when using Chrono to simulate contacts between

deformable objects as penetrations would occur. Severe penetrations, especially in simulations of shells or garments, often have a catastrophic effect in the outcome of the simulation due to mesh entanglement. This was observed already in simulations less complex than the ones we aim for (e.g. Fig. 1 and Fig. 4). As far as we can tell, Chrono does not provide strong guarantees of penetration-freeness for such cases of contact [6] which in turn rendered it unsuitable for our applications. Finally, systems based on the same simulation techniques than Stark have been discussed in the literature, e.g. [13]. However, actual open-source code has not been made available and evaluations of such systems in reproducing real-world complex scenes has not been presented.

In this landscape of simulation solutions, when available options are insufficient, researchers are left with the option of developing their own code or, most often, modifying other researchers' reference implementations. This typically leads to new or rehashed implementations tailored to specific types of problems with little regard for further development. For instance, the authors of the IPC method have provided reference implementations of their contact model in different contexts including volumetric deformables [12], co-dimensional objects [25] and rigid bodies [26] as well as releasing a toolkit [27] with a set of common functionalities to assist with the implementation of their method. These reference implementations have already been adopted by researchers to solve specific robotics problems. For example, Kim et al. [24] propose IPC-GraspSim that extends the original IPC reference implementation to model the specific problem of parallel-jaw grasping. While the conclusion of their studies is positive, as these new contact models do indeed outperform existing solutions, the community is now left with two different reference implementations: the original simulator that can handle contacts accurately, but does not support grippers out-of-the-box, and the new simulator that is effectively a fork specialized for a single type of problem.

Stark is designed to address this issue by providing a comprehensive platform for state-of-the-art techniques in simulation in the context of optimization time integration. Crucially, Stark is built on top of well-studied and tested simulation techniques and is itself, as a system, validated with real-world experiments. Additionally, Stark is designed to facilitate adding new components, such as controllers, and new materials, allowing researchers to test their models with ease in an rich featured environment.

III. CONCEPT

In this section, we very concisely highlight the most important concepts upon which our framework is built. We refer the reader to the foundational works of the individual components as they are topics of great complexity and it is beyond the scope of this paper to present them in detail.

A. Optimization Time Integration

We use an optimization time integration scheme in order to model the implicit time stepping of the simulation as a minimization problem. This has the advantage of allowing the use of robust, well-studied, minimization techniques which have stronger convergence guarantees than typical non-linear system solvers [28], [11], [12]. Let us first establish the relation between a potential energy source Ψ_i and its corresponding force $\mathbf{f}_i$ as

$$\mathbf{f}_i = -\mathbf{G}^T \frac{\partial \Psi_i}{\partial \mathbf{x}}, \tag{1}$$

where $\mathbf{G}$ is the kinematic map ($\dot{\mathbf{x}} = \mathbf{G}\mathbf{v}$) [29]. For particle systems and mesh discretizations potentials, $\mathbf{G}$ is the identity matrix. For rigid bodies, $\mathbf{x}$ is the generalized position vector which commonly is the concatenation of translations and rotations (as a quaternion) and $\mathbf{v}$ is the generalized velocity vector which contains the linear and angular velocities. For particle systems and mesh discretizations, $\mathbf{x}$ and $\mathbf{v}$ are the positions and velocities of the discretization points or mesh vertices. A set of values for the degrees of freedom $\mathbf{u}$ (typically positions or velocities) that fulfills the balance of forces is therefore a minimizer of the global potential energy

$$\sum_i \mathbf{f}_i(\mathbf{u}) = \mathbf{0} \longleftrightarrow \mathbf{u} = \underset{\mathbf{u}}{\operatorname{argmin}} \sum_i \Psi_i(\mathbf{u}), \tag{2}$$

which can be found using Newton's method

$$\frac{\partial^2 \Psi(\mathbf{u}^k)}{\partial \mathbf{u}^2}(\mathbf{u}^{k+1} - \mathbf{u}^k) = -\frac{\partial \Psi(\mathbf{u}^k)}{\partial \mathbf{u}}, \tag{3}$$

for a sequence $\mathbf{u}^{0\dots m}$ until the residual of the balance of forces is below a threshold $\|\frac{\partial \Psi(\mathbf{u}^m)}{\partial \mathbf{x}}\|_{\inf} < \epsilon$, which we set to $\epsilon = 1 \times 10^{-6}\,\mathrm{N}$ for the experiments shown in this paper.

The goal of Stark is to reliably reach accurate balance of forces in tough non-linear problems with potentially bad conditioning due to stiff non-linear materials or strong contacts. We use Newton's method due to the strong convergence guarantees and super-linear convergence close to the solution. Support for projections to positive-definiteness of the local Hessians is provided.

B. Rigid Bodies and Implicit Constraints

We use the inertial energy for rigid body dynamics as defined by Macklin et al. [30]:

$$\Psi_{RB} = \frac{1}{2}(\mathbf{v}^{n+1} - \hat{\mathbf{v}})^T \mathbf{M}^n (\mathbf{v}^{n+1} - \hat{\mathbf{v}}), \tag{4}$$

where $\mathbf{M}$ is the generalized mass matrix which contains the mass and the inertia tensors of the rigid bodies. The superscripts indicate the time step, $\hat{\mathbf{v}}$ is the inertial generalized velocity of the rigid bodies $\hat{\mathbf{v}} = \mathbf{v}^n + \Delta t \mathbf{a}^n$, where $\mathbf{a}^n$ is the generalized acceleration which includes linear accelerations and torque. Damping is also provided.

In order to allow for the composition of rigid body systems, Stark implements a collection of constraints, such as ball joints, hinges, sliders, dampers and motors, all of which are handled implicitly within the optimization time integration in order to unconditionally ensure stability. We point the reader to the Stark repository for a comprehensive description of all the joints, including how they are formulated. Further, an implicit controller is also provided which

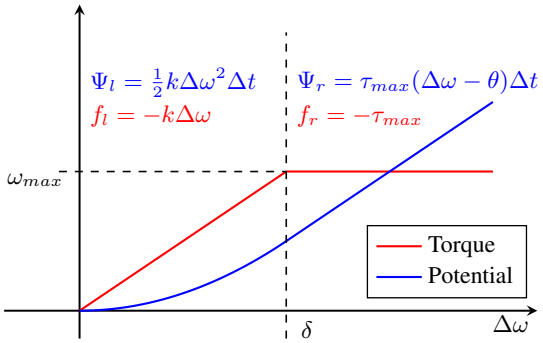

Fig. 2: Torque profile of an implicit motor controller provided in Stark. The input is $\Delta\omega$ which is the difference between the target and the current angular velocity of the bodies connected by the motor in the direction of the motor axis. There are two regimes separated by $\Delta\omega = \delta$: on the left, the application of the torque is progressive and proportional to how far from the target velocity the motor is and on the right, the maximum torque is applied. The rate of application $k = \frac{\tau_{max}}{\delta}$ and the offset $\theta = \frac{\delta}{2}$ are given by δ, which controls the sensitivity of the torque application.

enables smooth application of forces with limits. We employ this controller in the motors of the vacuum cleaning robot to smoothly maintain the robot's operational velocity while ensuring the motors respect the maximum stall torque given by the specifications, and in the gripper hand to ensure that it operates at a target close-in velocity but that cannot deliver more than a certain grip pressure. Fig. 2 shows the power profile of the implicit motor.

C. Deformable Solids with FEM

Stark uses non-linear hyperelastic constitutive models, e.g., Neo-Hookean, to model embedded (shells and cloths) and volumetric deformable objects. These material models can be included in the global formulation by using the Finite Element Method and integrating the strain energy density over the deformable objects. Bending is modeled with the method of Bergou et al. [31]. Using continuous models provides more controllability and higher fidelity than mass-spring systems or networks of chained rigid bodies. Together with the tight balance of forces that the Newton solver can achieve, Stark guarantees that material deformations are convergent under space and time refinement, which is an important property that many popular frameworks used in robotics do not provide. This approach has the important property of allowing to use finer meshes or to reduce the time step size of a simulation to improve the resolution of the material response without the need to find a new parametrization due to changes in the material behavior. Additionally, inertial- and strain-based damping is incorporated in order to improve the solution for materials that are more dissipative, such as cloth.

D. IPC Contact and Friction

Stark supports implicit frictional contact for rigid bodies and deformables, volumetric and embedded, including collisions between them, and self-collisions, which no other open-source simulation currently offer. We use the IPC contact and friction models [12] since they meet the robustness requirements of the applications Stark aims to provide solutions for. The high level idea of this contact approach is that when a pair of surface primitives (triangles, edges and points) are closer to each other than a very small user-defined contact distance $\hat{d}$, separation contact forces are applied. A crucial characteristic of the model is that the stiffness of the contact force response is adapted throughout the simulation to guarantee that enough force can be built up to resolve all the contacts in a given time step, ensuring, in turn, penetration-freeness. Intersection tests are carried out in the line search of Newton's method to guarantee that no invalid configuration is ever considered. In regards to friction, IPC proposes a smoothed unified potential based on Coulomb's friction that transitions between stick and slide within a small velocity threshold ϵ_v. While perfect sticking cannot be captured in this solution, the error induced by the smoothing can be severely reduced by choosing a small ϵ_v, at the cost of higher runtime.

IPC was originally developed in the context of computer graphics simulations but it has been already employed with success in the context of robotics [24].

IV. EXPERIMENTS

In this section, we first validate our simulator by reproducing real-world laboratory experiments of a mobile vacuum cleaner robot interacting with a towel on the floor in different configurations. Then, to demonstrate the general purpose nature of Stark, we show a simulation of a Franka robotic gripper grasping a deformable plastic cup. These simulations can be reproduced using the models included in Stark by default. See the attached video for the real-world experiments as well as the visualization of all simulations.

A. Validation of Robot-Cloth Interactions

We validate our simulator in a robot-cloth interaction simulation, a challenging yet relevant use-case driven by the vision of domestic robots that can better deal with everyday household environments.

We run 16 different variations of a vacuum robot driving towards a towel on the floor, in eight towel configurations and 2 different floor types: carpet (high friction) and polished (low friction). See Fig. 3 for each variation in simulation and the lab experiment. The goal is to evaluate whether the simulation can reproduce the qualitative outcome of each experiment, categorized as the robot pushing the towel (P), driving over the towel (O) or getting stuck (S). Note that success or failure does not depend on how geometrically similar lab and simulation results look at the end of each run, which is notoriously hard and not necessarily a good categorization for all applications. For the purpose of robot interaction and learning, we evaluate that the outcome is reproduced and that it happened due to the same mechanical interaction. The results for all the cases are shown in TABLE I.

In both, real-world experiments and simulations, we notice how the type of floor is a crucial factor that influences the

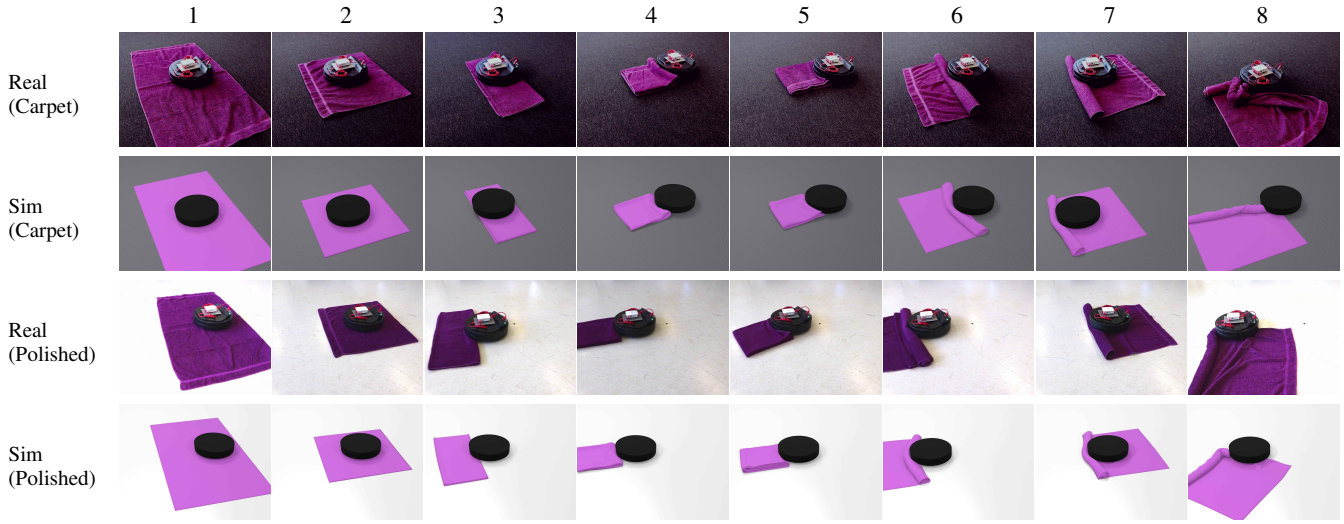

Fig. 3: Real and simulated outcomes of the 16 runs in the robot-towel interaction experiment. The columns correspond to the configurations 1 to 8 in TABLE I. Simulation results visualized with Blender's Eevee renderer.

	Carpet								Polished floor							
Towel config	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
Lab outcome	O	O	O	S	S	S	S	S	O	O	P*	P	P*	P	O*	S
Stark outcome	O	O	O	S	S	S	S	S*	O	O	P	P	P	P	O	S

TABLE I: Outcomes of the 16 setups of the robot-towel experiments, categorized as (P) robot pushing the towel, (O) driving over the towel or (S) getting stuck. Each lab experiment has been ran 5 times and the most frequent outcome is taken. Simulations are deterministic. In setup Polished-3, marked with *, the robot made it over the towel twice. In Polished-5 and -7 it got stuck twice. In setup Carpet-8 the simulation successfully replicates the real-world outcome (robot gets stuck due to the same underlying interactions) but there are discrepancies in the intermediate results as the towel doesn't bend.

outcome. While the robot cannot push the towel on the carpet floor in any instance due to the higher friction, it is able to push it in four occasions on the polished floor (cases 3 to 7). The robot can drive over the towel on the polished floor when the towel is more extended (cases 1 and 2) due to larger contact area with the floor and lower stacked thickness. Deformations also play an central role as they influence the robot's pose, lifting it from the front (cases 4 to 8 for both floor types) or providing uneven and changing resistance (case 8 for the polished floor). This has a direct effect in the wheels positioning and how much contact pressure there is, which relates to the available traction via friction. This, combined with the motor's power delivery, determines the robot's reaction and ultimately the outcome and how much energy is required during the interaction. This complex chain of interactions highlight the motivation to use strong coupling with tight residuals. Otherwise errors due to low tolerances quickly accumulate, severely reducing confidence in the simulation results. Despite the complexity of the models employed and their intricate interactions, Stark can reproduce the outcome of the experiments in 100% of the cases for the established criteria given a fixed parametrization of the floors, towel and robot across all simulations.

Here we list the most relevant parameters for the simulations. We model the robot as specified in the open-source Kobuki repository [32]. The body is approximated by a cylinder with radius $17.5\,\mathrm{cm}$ and height $7\,\mathrm{cm}$ although a better fit mesh is used for collision. The powered wheels have a radius of $3.5\,\mathrm{cm}$, while the radius of the non-powered front wheels is $1.35\,\mathrm{cm}$. The total mass of the robot is $2.951\,\mathrm{kg}$ and the clearance to the floor is $1.35\,\mathrm{cm}$. The robot is put together using joints provided by Stark. The free-spinning front and back wheels are attached to the body using hinge joints. A motor is used to attach each powered wheel to a suspension arm, which is in turn attached to the body by a damped spring. The motors deliver a maximum torque of $66.6\,\mathrm{mN\,m}$ (as in spec) and a maximum angular velocity of $7.43\,\mathrm{rad\,s^{-1}}$ to match the peak robot velocity of $0.26\,\mathrm{m\,s^{-1}}$.

The strain of the towel is modeled using FEM with the Neo-Hookean constitutive model and linear Lagrangian triangular elements. We use $E = 1.5{\times}10^4\,\mathrm{Pa}$ for the Young's modulus and $\nu = 0.3$ for Poisson's ratio. The low Young's modulus reproduces well that the cloth opposes nearly no resistance to in-plane compression. Additionally as fabric is almost inextensible, we employ edge-length based strain limiting to give increased resistance to overstretches beyond 10%. We use a bending stiffness ratio of $k_b = 5 \times 10^{-6}$. Without damping, the simulation would result in an almost perfectly elastic piece of soft rubber, far from the real-world cloth behavior which is very energy dissipative. We use Rayleigh damping with an inertial damping factor of 2 and a 0.1 ratio relative to the stiffness for both the strain

Fig. 4: Simulation of a Franka Emika Hand grasping a plastic cup with weights. Results visualized with Blender's Cycles path tracer.

and bending damping. The total mass of the towel, which is lumped to the nodes, is $0.484\,\mathrm{kg}$ and its dimensions are $1.75\,\mathrm{m}$ by $0.75\,\mathrm{m}$. All the parameters are set based on hands-on experimentation with the real towel, such as waviness when held by one corner or stack height when folded. The parametrization is transferable to other simulations using the same material and frictional contact models.

The thickness of the towel is $3.25\,\mathrm{mm}$, which we conveniently use as the contact distance of $\hat{d}$. We use the a friction of $\mu_t = 0.4$ for the towel self-contacts, $\mu_k = 1.0$ for the robot-towel and -floor contacts and $\mu_c = 1.0$, $\mu_p = 0.2$ for the towel-floor contacts, carpet and polished, respectively. The friction parameters are used as the degrees of freedom of this experimentation setup and fit to match the simulation outcomes to the real experiments. All parameters have been kept constant over the entire experiment. The total runtime for the 16 simulations is roughly 6 hours on a AMD Ryzen Threadripper PRO 5975WX CPU with 32 cores @ $3.60\,\mathrm{GHz}$.

B. Grasping of a Plastic Cup

We also showcase a grasping simulation to demonstrate that Stark can be used in other challenging scenarios, such as grasping of deformable objects (see Fig. 4). Although not compared to real-world experiments, the simulation follow the expected results dictated by static friction analysis.

We model the parallel-jaw gripper Franka Emika Hand according to the manufacturer specifications [33]. We employ Stark's implicit joint controller to model the gripper with a target closing velocity of $100\,\mathrm{mm\,s^{-1}}$ and a maximum grip force of $5\,\mathrm{N}$. The contact distance for this simulation is $\hat{d} = 0.5\,\mathrm{mm}$ and the friction between the gripper and the cup is $\mu = 0.5$. The plastic cup, approximately $10\,\mathrm{cm}$ tall, uses the same shell/cloth models than the towel from the previous experiment but significantly stiffer in order to faithfully represent the behavior a of thin PET plastic sheet (see also the experiment's source code in Stark's repository for further details).

First, the jaws close to grab the plastic cup (Fig. 4 left), which is then deformed. Once the gripper and the deformed cup come to an equilibrium, steel balls of $14\,\mathrm{g}$ each are poured into the cup. Static grasping breaks when 20 balls are inside the cup (Fig. 4 center), as the vertical pull reaches $2.8\,\mathrm{N}$, surpassing the theoretical $2.5\,\mathrm{N}$ static friction threshold given by the $5\,\mathrm{N}$ contact pressure and the $\mu = 0.5$ Coulomb's friction. Due to the large deformations of the plastic shell and the reinforcement at the cup's lips, the cup is held for longer as it latches onto the jaws (Fig. 4 right). It takes 60 minutes to compute the 16 seconds of simulation on the aforementioned hardware, mostly due to the simulation length and amount of collisions between spheres. The simulation of the grasp itself, which happens in the first 2 seconds of the simulation, takes 5 minutes. This type of simulations, which are relevant to a wide range of robotic grasping tasks such as bin picking, are notoriously difficult since they require accurate and robust methods for both materials and contacts, and good coupling between the deformable and rigid body systems.

V. CONCLUSIONS

In this paper we presented Stark, a framework to simulate strongly coupled mechanical systems, including rigid and deformables, with frictional contact. The goal of our system is to incorporate recent advances in simulation to provide not only a larger set of supported phenomena than the most used frameworks currently available, but also a higher degree of modeling accuracy and robustness. We demonstrate the capabilities of Stark in simulations featuring non-linear materials undergoing large deformations, rigid body systems with implicit joints and motors, and complex contact configurations with friction. Stark simulations are able to match the outcome of all 16 simulations carried out in a lab setting where a vacuum cleaning robot interacts with a towel on the floor. Available simulation solutions lacked the required features to model such simulations or did not have the required accuracy, especially when handling frictional contacts with deformable objects. Stark is therefore posed to fill a gap in the current landscape of available simulation software and empower researchers with easy access to state-of-the-art simulation solutions.

As for limitations, Stark inherits the conditionals of the underlying models it uses, the most prominent one being the imposition for the models to have a variational form, which makes dissipative effects such as friction or damping more challenging to account for. Besides, there are paths for improvements within Stark. First, introducing additional mechanical effects, like plasticity for instance, is an interesting direction. Second, although the present optimizer implementation has shown its effectiveness in the provided examples, refining its performance would be beneficial as it is the current bottleneck of the simulation execution. Finally, adopting standard scene description files would further bolster its adoption.

ACKNOWLEDGEMENTS

We would like thank to Rupert Kinzler for his knowledgable advice in the development of Stark. We also thank Tobias Mauk for his support in carrying out the real-world experiments with the vacuum cleaner robot.

REFERENCES

[1] C. K. Liu and D. Negrut, "The role of physics-based simulators in robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, no. 1, pp. 35–58, 2021.

[2] R. L. Smith, "Ode: Open dynamics engine," https://www.ode.org, accessed on 10 September 2022.

[3] E. Coumans and Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning," http://pybullet.org, accessed on 10 September 2022.

[4] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012.

[5] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "DART: dynamic animation and robotics toolkit," *Journal of Open Source Software*, vol. 3, no. 22, 2018.

[6] A. Tasora, R. Serban, H. Mazhar, A. Pazouki, D. Melanz, J. Fleischmann, M. Taylor, H. Sugiyama, and D. Negrut, "Chrono: An open source multi-physics dynamics engine," in *High Performance Computing in Science and Engineering*, T. Kozubek, Ed. Springer, 2016.

[7] R. L. Smith, "Ode: Open dynamics engine," https://developer.nvidia.com/physx-sdk, accessed on 10 September 2022.

[8] J. Bender, M. Müller, M. A. Otaduy, and M. Teschner, "Position-based methods for the simulation of solid objects in computer graphics." in *Eurographics (State of the Art Reports)*, 2013, pp. 1–22.

[9] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly, "Projective dynamics: Fusing constraint projections for fast simulation," *ACM Trans. Graph.*, vol. 33, no. 4, 2014.

[10] L. Lan, M. Li, C. Jiang, H. Wang, and Y. Yang, "Second-order stencil descent for interior-point hyperelasticity," *ACM Trans. Graph.*, vol. 42, no. 4, 2023.

[11] T. F. Gast, C. Schroeder, A. Stomakhin, C. Jiang, and J. M. Teran, "Optimization integrator for large time steps," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 10, 2015.

[12] M. Li, Z. Ferguson, T. Schneider, T. Langlois, D. Zorin, D. Panozzo, C. Jiang, and D. M. Kaufman, "Incremental potential contact: Intersection-and inversion-free, large-deformation dynamics," *ACM Trans. Graph.*, vol. 39, no. 4, 2020.

[13] Y. Chen, M. Li, L. Lan, H. Su, Y. Yang, and C. Jiang, "A unified Newton barrier method for multibody dynamics," *ACM Trans. Graph.*, vol. 41, no. 4, 2022.

[14] J. A. Fernández-Fernández, F. Löschner, L. Westhofen, A. Longva, and J. Bender, "Symx: Energy-based simulation from symbolic expressions," *arXiv preprint arXiv:2303.02156*, 2023.

[15] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2004.

[16] Cyberbotics Ltd., "Webots," http://www.cyberbotics.com, accessed on 10 September 2022.

[17] Coppelia Robotics, Ltd., "CoppeliaSim," https://www.coppeliarobotics.com, accessed on 10 September 2022.

[18] Nvidia Corporation, "Isaac Sim," https://developer.nvidia.com/isaac-sim, accessed on 10 September 2022.

[19] M. Körber, J. Lange, S. Rediske, S. Steinmann, and R. Glück, "Comparing popular simulation environments in the scope of robotics and reinforcement learning," 2021, preprint arXiv:2103.04616 [cs.RO].

[20] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: comparison of Bullet, Havok, MuJoCo, ODE and PhysX," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015.

[21] Y. Bai and C. K. Liu, "Coupling cloth and rigid bodies for dexterous manipulation," in *7th Int. Conf. on Motion in Games*. New York, NY, USA: Association for Computing Machinery, 2014.

[22] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[23] I. Huang, Y. Narang, C. Eppner, B. Sundaralingam, M. Macklin, R. Bajcsy, T. Hermans, and D. Fox, "DefGraspSim: Physics-based simulation of grasp outcomes for 3d deformable objects," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, 2022.

[24] C. M. Kim, M. Danielczuk, I. Huang, and K. Goldberg, "IPC-GraspSim: Reducing the sim2real gap for parallel-jaw grasping with the incremental potential contact model," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2022.

[25] M. Li, D. M. Kaufman, and C. Jiang, "Codimensional incremental potential contact," *ACM Trans. Graph.*, vol. 40, no. 4, jul 2021. [Online]. Available: https://doi.org/10.1145/3450626.3459767

[26] Z. Ferguson, M. Li, T. Schneider, F. Gil-Ureta, T. Langlois, C. Jiang, D. Zorin, D. M. Kaufman, and D. Panozzo, "Intersection-free rigid body dynamics," *ACM Trans. Graph.*, vol. 40, no. 4, 2021.

[27] Z. Ferguson *et al.*, "IPC Toolkit," 2020. [Online]. Available: https://ipc-sim.github.io/ipc-toolkit/

[28] M. Ortiz and L. Stainier, "The variational formulation of viscoplastic constitutive updates," *Computer Methods in Applied Mechanics and Engineering*, vol. 171, no. 3, 1999.

[29] J. Bender, K. Erleben, and J. Trinkle, "Interactive simulation of rigid body dynamics in computer graphics," *Comput. Graph. Forum*, vol. 33, no. 1, 2014.

[30] M. Macklin, K. Erleben, M. Müller, N. Chentanez, S. Jeschke, and T. Y. Kim, "Primal/dual descent methods for dynamics," in *ACM SIGGRAPH/Eurographics Symp. on Computer Animation*. Goslar, DEU: Eurographics Association, 2020.

[31] M. Bergou, M. Wardetzky, D. Harmon, D. Zorin, and E. Grinspun, "A Quadratic Bending Model for Inextensible Surfaces," in *Symposium on Geometry Processing*, A. Sheffer and K. Polthier, Eds. The Eurographics Association, 2006.

[32] Yujin Robot, "Kobuki description model," https://github.com/yujinrobot/kobuki_description, accessed on 10 September 2022.

[33] Franka, "Franka Hand: Product Manual," https://download.franka.de/documents/220010_Product%20Manual_Franka%20Hand_1.2_EN.pdf, 2022, [Online; accessed Sept 2023].