Robotics: Science and Systems 2023
Daegu, Republic of Korea, July 10-July 14, 2023

Demonstrating RFUniverse: A Multiphysics Simulation Platform for Embodied AI

Haoyuan Fu*
Shanghai Jiao Tong University
simon-fuhaoyuan@sjtu.edu.cn

Wenqiang Xu*
Shanghai Jiao Tong University
vinjohn@sjtu.edu.cn

Ruolin Ye*
Cornell University
ry273@cornell.edu

Han Xue
Shanghai Jiao Tong University
xiaoxiaoxh@sjtu.edu.cn

Zhenjun Yu
Shanghai Jiao Tong University
jeffson-yu@sjtu.edu.cn

Tutian Tang
Shanghai Jiao Tong University
tttang@sjtu.edu.cn

Yutong Li
Shanghai Jiao Tong University
davidliyutong@sjtu.edu.cn

Wenxin Du
Shanghai Jiao Tong University
mnkmYuki@sjtu.edu.cn

Jieyi Zhang
Shanghai Jiao Tong University
yi_eagle@sjtu.edu.cn

Cewu Lu
Shanghai Jiao Tong University
lucewu@sjtu.edu.cn

***Abstract*—Multiphysics phenomena, the coupling effects involving different aspects of physics laws, are pervasive in the real world and can often be encountered when performing everyday household tasks. Intelligent agents which seek to assist or replace human laborers will need to learn to cope with such phenomena in household task settings. To equip the agents with such kind of abilities, the research community needs a simulation environment, which will have the capability to serve as the testbed for the training process of these intelligent agents, to have the ability to support multiphysics coupling effects.**

Though many mature simulation software for multiphysics simulation have been adopted in industrial production, such techniques have not been applied to robot learning or embodied AI research. To bridge the gap, we propose a novel simulation environment named RFUniverse. This simulator can not only compute rigid and multi-body dynamics, but also multiphysics coupling effects commonly observed in daily life, such as air-solid interaction, fluid-solid interaction, and heat transfer.

Because of the unique multiphysics capacities of this simulator, we can benchmark tasks that involve complex dynamics due to multiphysics coupling effects in a simulation environment before deploying to the real world. RFUniverse provides multiple interfaces to let the users interact with the virtual world in various ways, which is helpful and essential for learning, planning, and control. We benchmark three tasks with reinforcement learning, including food cutting, water pushing, and towel catching. We also evaluate butter pushing with a classic planning-control paradigm. This simulator offers an enhancement of physics simulation in terms of the computation of multiphysics coupling effects. The simulation environment, videos, and other supplementary materials can be viewed on the website: https://sites.google.com/view/rfuniverse.

I. INTRODUCTION

Physics laws govern the real world and therefore generate various object dynamics. Multiphysics phenomena can happen when objects in different states interact with each other. Humans can encounter many these common phenomena when conducting everyday household tasks. For instance, a piece of towel dropping in the air can easily drift its path due to *aerodynamics*. The water from a faucet can flush away the food waste and dirts through *fluid-solid interaction*. The *heating* process can soften the butter and transfer its state from solid to liquid. An intelligent agent which learns to perform household tasks will need to learn to react with object state changes due to these common multiphysics coupling effects.

However, though many simulation environments [17, 44, 3, 29, 33, 25, 45, 32, 19, 9, 6, 13, 46, 48] have been developed for embodied AI research in recent years, they usually focus on tasks which only require the basic rigid and multi-body dynamics simulation, such as abstract decision-making or

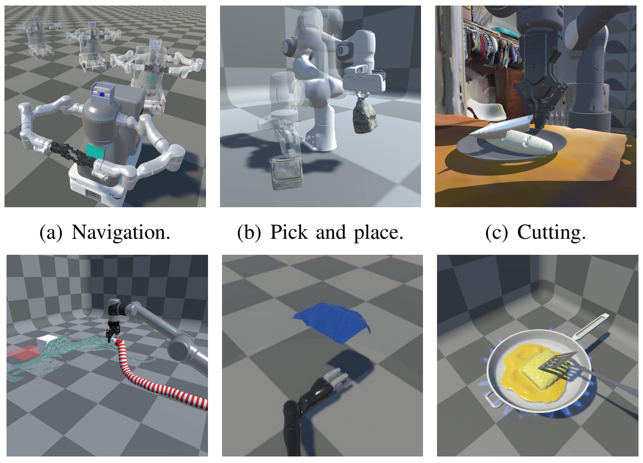

(a) Navigation. (b) Pick and place. (c) Cutting.
(d) Water pushing. (e) Towel catching. (f) Butter pushing.

Fig. 1. Tasks in RFUniverse: Standard tasks such as (a) Navigation (b) Pick and place, and tasks involving multiphysics interaction such as (c) Physics-based cutting; (d) Water pushing; (e) Towel catching; (f) Butter pushing.

* Denotes equal contribution. Cewu Lu is the corresponding author, and member of Qing Yuan Research Institute and MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China. Ruolin Ye was with Shanghai Jiao Tong University when work was done.

simple manipulation. Almost none of them considers the object dynamics under coupled physics effects. A recent simulation environment OmniGibson [20] supports over 2K household tasks, which covers the largest scope of household tasks by far. However, even though it contains tasks that involve complex physics coupling effect, like water pouring and gas ignition, the underlying multiphysics interaction is still simplified or ignored in this environment. These kinds of simplification limit the task scope of embodied AI research and limits the sim-to-real transfer ability of the algorithms developed in simulation.

To bridge the gap and extend the task scope of embodied AI research, we present a novel simulation environment named **RFUniverse**. Aside from common rigid and multi-body dynamics simulation, RFUniverse integrates three kinds of multiphysics coupling effects into the simulation environment, namely air-solid interaction (aerodynamics), fluid-solid interaction (hydrodynamics), and heat transfer (thermodynamics). Without a doubt, in nature, different physics laws have more than these three ways of coupling to influence the world. However, it is not practical to integrate all kinds of multiphysics effects into the simulation while maintaining a real-time performance which is essential for robot learning and control. Therefore, we choose to implement these three multiphysics effects which are the most commonly encountered in our daily life and therefore will have the potential to better support common household tasks. With the underlying support of multiphysics, there will be chances for the research community to evaluate the execution process for advanced physics-based tasks. We focus on four key tasks in this paper, including physics-based **food cutting**, **water pushing**, **towel catching**, and **butter pushing**.

Admittedly, a high-accuracy calculation of multiphysics interaction [26, 35] requires considerable computational resources so that is difficult to run in real time, making it not suitable for the real-time requirement from the robotics community. To balance usability and accuracy, we adopt solutions with simplification on the physics side with a decent computation speed while maintaining fidelity to real-world dynamics. To prove the fidelity, we conduct parallel experiments to compare the multiphysics effects in the virtual world with the ones in the real world. For example, with a glass of water, we observe its weight and volume and find out it is the same in the real world. We fix a towel and blow it with the wind, finding the virtual world and the real one are quite aligned in terms of the movement of the towel. We heat butter and see it melting down similarly in simulation and the real world. We hope our simulation environment with verified fidelity can serve as a platform for multiphysics-based robot learning, and facilitate the integration of advanced simulation techniques into embodied AI research.

To help utilize the simulation environment, in addition to the multiphysics simulation, RFUniverse also provides full functionalities to support task simulation and learning: physics-based rendering, multi-modal sensing, Python APIs, and a gym-like wrapper for reinforcement learning. Besides, we provide a ROS-free version of MoveIt [4], RFMove [49], as a lightweight planner, and natively integrate it into RFUniverse for full-body movement planning. In addition, we also provide ROS interface to enlarge the ecosystem and expand the scope of potential users. We also provide a VR interface to extend the interactive ability from the real to the simulated environment.

We present a set of standard tasks trained with reinforcement learning to showcase the usage of RFUniverse to robot learning in supplementary materials, such as navigation in the setting of multi-agent collaboration, locomotion in the setting of goal reaching, and a few manipulation tasks (*e.g.* fruit picking, cloth folding, sponge wiping). In the main paper, we would like to highlight and focus more on introducing and explaining the multiphysics-based system, which is seldom considered by previous simulation environments. We evaluate food cutting, water pushing, towel catching with reinforcement learning, and butter pushing with classic planning control, to demonstrate the usage of this multiphysics-based system. We also perform experiments for visual pre-trained encoders for reaching and cabinet closing.

We summarize our contributions as follows:

- We propose a multiphysics-based simulation environment RFUniverse for embodied AI research. RFUniverse provides a client-server communication framework based on gRPC, which can enable full functionality control of Unity with multiple interfaces. It provides physics-based rendering and multi-modal sensing, enabling the agent to perceive the physics information in the virtual world.
- With consideration of the balance between fidelity and computational cost, RFUniverse takes the first step towards building a simulation platform for manipulation tasks involving multiphysics dynamics in robot manipulation, including structural mechanics, aerodynamics, hydrodynamics, and thermodynamics.
- We conduct extensive parallel experiments and prove the fidelity of the real-time simulation by comparing them with the real world in the scope of tasks involving multiphysics phenomena, and verify the fidelity of the physics computation in the simulated environment.
- With RFUniverse, we extend the task scope to multiphysics-based tasks, which have rarely been explored before. With the extended task scope, we are able to benchmark three tasks with reinforcement learning, namely physics-based cutting, water pushing, and towel catching; we benchmark one task with planning and control for butter pushing.

II. Related Works

A. *Multiphysics Simulation Software*

In the field of modern engineering and science research, the problems left to solve usually require solutions that span a multitude of physical phenomena. There are a lot of commercial (*e.g.* COSMOL [26], Abaqus [35]) or open-source (*e.g.* MOOSE [22]) softwares designed for such demands. However, due to the requirement for high-accuracy, the algorithms behind these softwares prefer to take a lot of time and

TABLE I

Environment	Physics	Learning	Rendering	Integrated Planner	VR	ROS
COSMOL [26]	aero, hydro, thermo, multi-body	/	/	/	/	/
Abaqus [35]	aero, hydro, thermo, multi-body	/	/	/	/	/
Chrono [40]	hydro, multi-body	/	/	/	/	/
AGX Dynamics [1]	aero, hydro, multi-body	✓	photo-	✓	/	/
HoME [3]	multi-body	✓	photo-	✓	/	/
AI2THOR series [17, 6, 9]	multi-body	✓	photo-	/	/	/
Habitat [25]	multi-body	✓	photo-	/	/	/
SAPIEN [46]	multi-body	✓	physics-	✓	✓	/
VirtualHome [29]	multi-body	✓	photo-	/	/	/
VRKitchen [13]	multi-body	✓	photo-	/	/	✓
ThreeDWorld [11, 12]	multi-body	✓	photo-	/	✓	/
Gibson series [44, 45, 32, 19, 20]	multi-body	✓	physics-	✓	✓	✓
Isaac Gym [24]	multi-body	✓	physics-	/	/	/
RFUniverse	aero, hydro, thermo, multi-body	✓	physics-	✓	✓	✓

Comparison with different simulation environments. As some environments are developed in a long run as different versions, we summarize them into a series and show the features based on the latest version.
Physics: "aero" for aerodynamics, "hydro" for hydrodynamics, "thermo" for thermodynamics, "multi-body" for multi-body dynamics. We check if it has at least one function that involves such dynamics. To note, some simulators exhibit water rendering, but such fluid cannot provide interaction force.
Learning: whether support reinforcement learning or support to produce a visual dataset for perception tasks.
Rendering: "photo-" for photorealistic rendering effect, "physics-" for physics-based rendering effect (at least ray-tracing enabled).

computational resources to process, which make them hard to be integrated into a learning pipeline, where the simulation results should be produced on-the-fly.

Among these softwares, only a few provide interfaces for robotics, namely Project Chrono [40], and AGX dynamics [1]. And only AGX dynamics supports real-time simulation. From the multiphysics perspective, AGX dynamics supports fluid mechanics and aerodynamics. The solid mechanics in AGX dynamics are limited to the APIs it provides.

In our solution, we manage to integrate different physics backends together, namely, PhysX [27], SOFA [10], Obi [38], Zibra [15]. These physics backends do not provide robot interfaces on their own, but they have specialties to simulate different physics phenomena.

B. Physics-based Simulation in Embodied AI

In the past years, we have witnessed a rapid growth of simulation environments for embodied AI or robot learning. Most of the simulation environments adopt one physics engine for physics simulation, such as PhysX [27], MuJoCo [41], Bullet [5], Flex [21], ODE [36] and so on. Most of these physics engines can provide high-fidelity rigid and multi-body dynamics simulation. However, not a single physics engine can support all the multiphysics simulations. The limitation of simulation also setback the development of robot learning on some advanced manipulation tasks such as deformable object manipulation, arbitrary object cutting (i.e. not slicing the object in advance), and those involving object state change due to multiphysics interaction. As a result, though we already have many simulation environments available [17, 44, 3, 29, 33, 25, 45, 32, 19, 9, 6, 13, 46, 48], among them, a most recent simulation environment OmniGibson [20] comprises of the most comprehensive household tasks so far. They all fail to consider the multiphysics tasks.

A detailed comparison between RFUniverse and previous environments can be referred to in Table I.

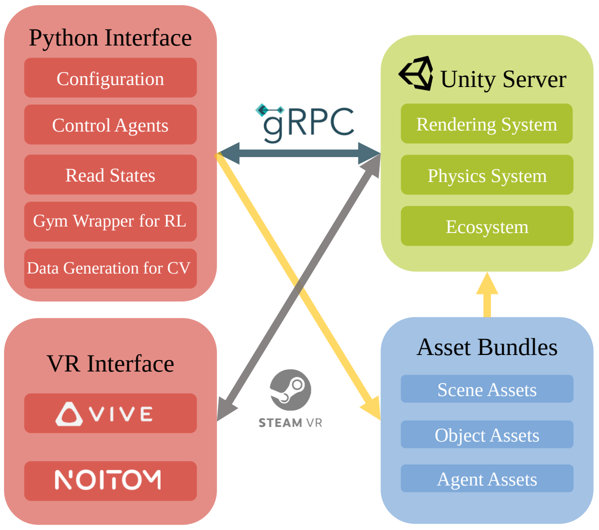


Fig. 2. The framework of RFUniverse.

III. RFUniverse Simulation Environment

RFUniverse is a simulation environment that provides high-fidelity physics simulation for multiphysics interaction. Based on a server-client interface, it supports multiple interfaces to interact with the Unity server, which integrates different physics backends for various object types, thus enabling the simulation of multiphysics coupling effects.

A. Communication Framework & Interfaces

RFUniverse adopts a server-client framework based on gRPC, enabling Python and VR interfaces to communicate with the Unity backend. We show its structure in Fig. 2. This communication framework supports multiple languages and OS platforms. We adopt Python as the interfacing language, which is widely used in learning frameworks [28, 16], due to its simplicity and ecosystem. Different VR devices can be connected to RFUniverse through SteamVR. We demonstrate the VR interface with the HTC Vive headset and Noitom Hi5 glove in Fig. 3.

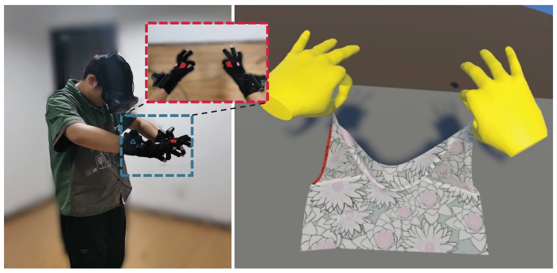


Fig. 3. The VR interface with HTC Vive and Noitom Glove.

B. Multiphysics Interaction

Object state in daily life is governed by different physics laws. Conventionally, these physics phenomena are studied by different branches of mechanics, such as solid mechanics, fluid mechanics (hydrodynamics), aerodynamics, and thermodynamics. Despite the fact there are mature commercial softwares for precise computation of these mechanics during multiphysics interaction, they usually cost a decent amount of computational resources and time, making it hard for them to satisfy the need for real-time computation from the robotics side, and therefore seldom considered in embodied AI. To take the first step towards building a simulation environment for the manipulation tasks involving multiphysics dynamics, we adopt the solvers that balance physics fidelity and computational efficiency and integrate them in Unity.

In Unity, we borrow the wisdom from modern physics engines including [10] and [15] which support mesh-based and particle-based solvers. To be specific, for solid mechanics, we model the computation process with finite element mechanics with different constitutive models. By treating the object as a volume instead of a 2-manifold surface, it can support cutting the object from arbitrary positions in a physics realistic way. For hydrodynamics, aerodynamics, and thermodynamics, we leverage particle-based methods to simulate the diffusion process of fluid, air, and heat.

C. Physics-based Rendering System

Visual input is one of the important modalities for agents to perceive the world. Simulating a high-fidelity physics-based rendering system can mitigate the sim-to-real gap when deploying the algorithms from simulation to the real world. Though many previous simulation environments claim their simulators can provide "photorealistic" rendering effects, we realize such photorealism might be unfaithful in physics. As shown in Fig. 4, a "photorealistic" rendering pipeline cannot produce correct refraction effects for a glass and the liquid in a glass. In comparison, RFUniverse supports ray-tracing techniques for rendering and therefore is capable of handling complex optical effects in real-time. It can be rendered at 55 fps on an Nvidia RTX 3090 graphics card.

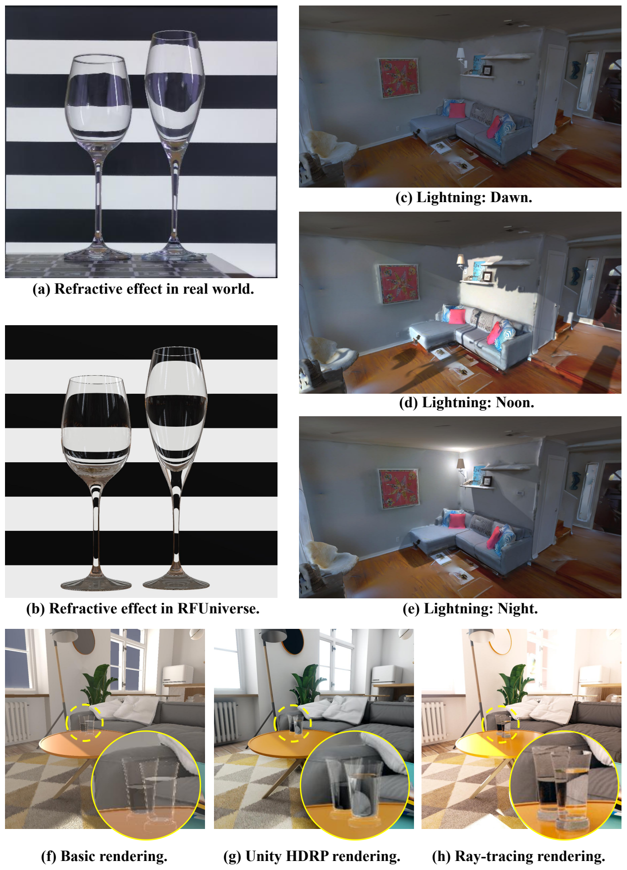
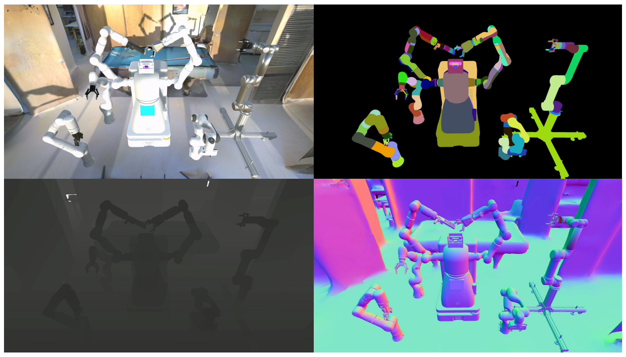
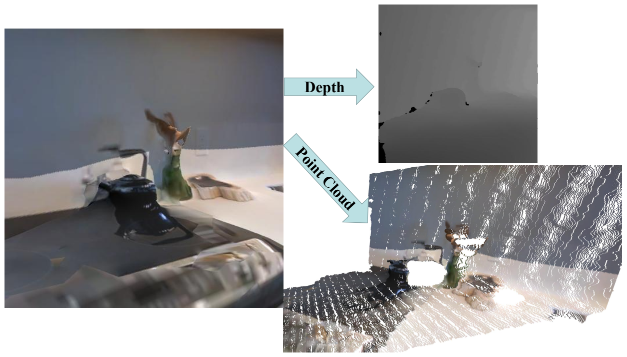

(a) Refractive effect in real world.
(b) Refractive effect in RFUniverse.
(c) Lightning: Dawn.
(d) Lightning: Noon.
(e) Lightning: Night.
(f) Basic rendering.
(g) Unity HDRP rendering.
(h) Ray-tracing rendering.

Fig. 4. **(a)(b):** Refractive effect in real world and in RFUniverse. Two wine glasses with gray code behind them. Here we mainly compare the refraction effects because it can reveal the physics fidelity of the rendering system; **(c)-(e):** different indoor lighting conditions; **(f)-(h):** Liquid in refraction rendering effects under basic rendering, 'photorealistic' rendering claimed by other simulation environments, and ray-tracing (physics-based) rendering.

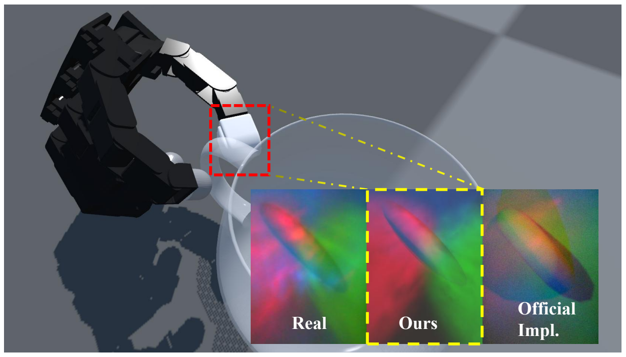
D. Multi-modal Sensing

To interact with the environment, the agent needs to have multi-modal sensory information. RFUniverse supports a set of physics realistic sensors to equip the agent with the capability of perception. For visual input, we leverage ray-tracing techniques and integrated the IR-based depth rendering proposed in Sapien [50] to RFUniverse, which mimics the sensor noise of IR-based depth sensors like RealSense D415 camera. Besides, we also notice the trending of vision-based tactile sensing research in the robotics community [34, 8, 42, 39, 37]. We re-implement and improve the DIGIT sensing [18] from TACTO [43], and provide a simulated vision-based DIGIT tactile sensing with a less sim-to-real gap. In Fig. 5(c), we place a mug which has a real-world copy into the scene. We then can compare our implementation of DIGIT sensing in RFUniverse with the original ones in TACTO. For force sensing, RFUniverse also has force/torque sensors in the robot joints as shown in Fig. 5(d).


(a) Visual sensor.



(b) IR-based depth sensor.



(c) Tactile sensor.

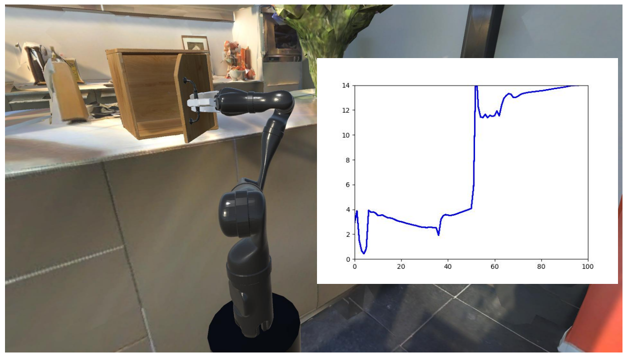

(d) Force sensor.

Fig. 5. Multi-modal Sensor. (a) Visual sensor: RGB, Instance mask, Perfect depth and Normal map; (b) IR-based depth sensor; (c) DIGIT Tactile Sensor with tactile image. We also compare our implementation with real-world DIGIT sensor and official implementation in Pybullet to show that the gap between DIGIT in real-world in RFUniverse is significantly smaller. For the real-world one, we take the image in the same setting.; (d) Force/Torque Sensor.

E. Assets

Our assets include objects and agents. The simulation environment supports a wide range of object types. For example, we have **.obj**, **.ply**, **.stl** file format for rigid and softbody object, **URDF** and **FBX** for articulated object, **vtk** for volumetric objects used for FEM simulation. We provide a ready-to-use full set of PartNet-Mobility [46], AKB-48 model repository [23], and Google Scanned dataset [7]. For robots, we support the most common robot arms, such as Franka Emika Panda, and UR5, and mobile robots including Stretch and Fetch, dual-arm mobile manipulator, for instance, PR2. As for the robot hand, we support Robotiq85, Barrett Hand, Allegro, Schunk 5-finger hand, and Shadow hand. If users want to use their custom objects or agents, they can simply load them on the fly via GUI or Python API.

F. Lightweight ROS-free Motion Planner

In many cases, an easy-to-use lightweight planner can help with prototyping and performing experiments. We modify Moveit! [4] to a lightweight ROS-free version, RFMove [49]. It can utilize the OMPL library for motion planning, and support obstacle avoidance by synchronizing the simulation scene and planning scene through Python API. We integrate it into the simulation environment to help alleviate the pain of a heavy stack when trying to use libraries for planning.

G. Gym-like Wrapper for Reinforcement Learning

As for reinforcement learning, we provide a standard gym-like wrapper for different kinds of environments. It supports multi-thread parallelization, which means users can train multiple agents simultaneously. We will later detail the RL experiments in Sec. IV, and we leave the basic manipulation experiments in supplementary materials.

IV. Experiments

To verify the fidelity of RFUniverse, we perform three experiments involving multiphysics coupling effects for hydrodynamics, aerodynamics, and thermodynamics in Sec. IV-B and compare them with the ones in the real world. We perform reinforcement learning for manipulation tasks involving multi-physics coupling effects in Sec. IV-C. We showcase the usage of our lightweight planner for melted butter manipulation task in Sec. IV-D. We also perform experiments for visual pre-trained encoders with reinforcement learning.

A. Machine Specification

During training, all experiments are benchmarked on Ubuntu 22.04 platform with Intel(R) Core(TM) i9-10900K CPU and 1 NVIDIA Geforce GTX 1080Ti graphic card. Due to different calculating burdens in the tasks, we dynamically adjust the action time step. We set the action time step for rigid object interaction $t_a^r = \frac{1}{50}s$, for deformable object, fluid, and air interaction $t_a^d = \frac{1}{25}s$, and for cutting task $t_a^c = \frac{1}{20}s$.

B. Physics Simulation Verification Experiments

a) Water Weight: We place a cup onto a weighing scale as shown in Fig. 6 (a). Then, we slowly add the water to the cup. Since the volume (V_w) and mass of water (m_w) satisfy $m_w = \rho \cdot V_w$, where ρ is considered as density and is calibrated from real-world, we can verify the weight reading by checking the volume of the water in the cup.

b) Towel in Wind: In the real world, we fix a towel to a pole and set up a hair dryer with different levels of wind aside. We record a video of it, and set up the same setting with similar wind conditions. We observe the video and find out they have similar trends.

c) Heat Transfer: We take a video clip of a human pushing a slice of butter on a heated skillet and imitate the pushing behavior from the video in simulation. We can observe the simulated butter is melted down in a similar way.

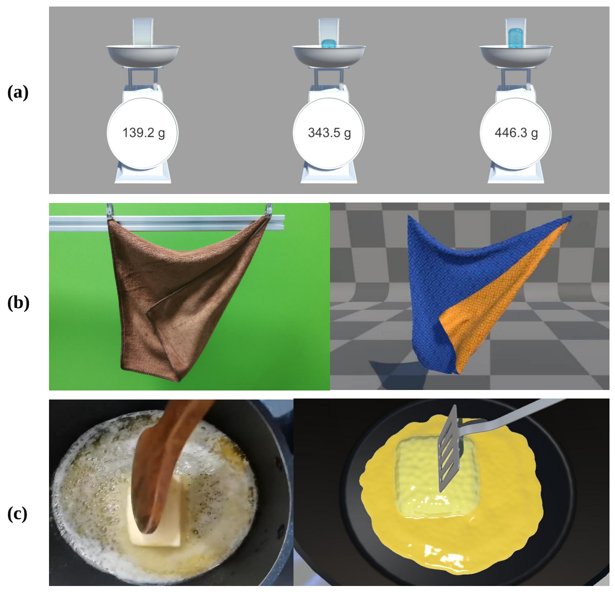


Fig. 6. Physics simulation verification experiment setting. (a) depicts water force in RFUniverse. Water is modeled with particles and each particle weighs about $1g$. The left image contains an empty glass that weighs around $140g$, while the glass in the middle image contains 200 particles and the glass in the right image contains 300 particles; (b) contains two cloth swaying in the wind in the real world and RFUniverse, which present a similar configuration; (c) contains two butter-melting scenes in the real world and RFUniverse.

C. Manipulation experiments with Reinforcement Learning

1) Physics-based Cutting:

a) Task Description: A banana is placed on the ground, and a Flexiv robot arm with an AG-95 gripper holds a knife in its hand. In each episode, the robot will reset to a fixed pose and the task is divided into two phases. In the first phase, a goal pose of the knife is given and the Flexiv robot arm should move and make the knife-in-hand closer to the given pose. This pose is sampled on the top surface of the banana with proper orientation. Then in the second phase, the Flexiv robot will move along its end-effector's Z-axis to cut the banana.

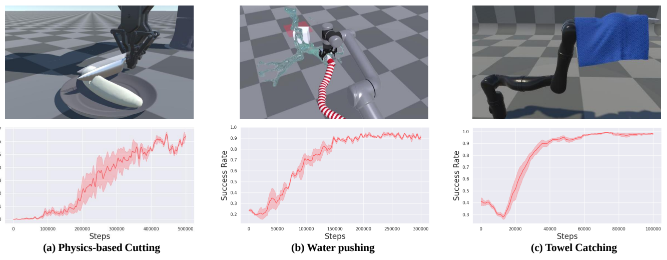


Fig. 7. Experimental results for manipulation experiments with reinforcement learning. **Upper:** The snapshots of task scenes. **Bottom:** The relationship between success rate and training step for manipulation tasks are displayed.

b) Task Implementation: In each step, a 6-d action is needed with the first 3 dimensions representing the delta translation of the gripper along the X, Y, and Z axis and the last 3 dimensions representing delta Euler angles along X, Y, and Z axis. The observation includes the pose and velocity of the end-effector, the pose of knife-in-hand, and the goal knife pose. All orientations in observation are in quaternion format. We use a two-stage sparse reward function in this task: the default reward is 0, and if the distance between knife-in-hand and goal knife is less than $4cm$, the reward will add 0.5; if the minimal angle between knife-in-hand and goal knife is less than 5 degree, the reward will add 0.5. The task is regarded as successful when the reward is 1. Since this is a goal-conditioned RL task, we use SAC [14] algorithm with HER [2] replay buffer to train the policy for this task.

2) Water Pushing:

a) Task Description: A fixed-base UR5 robot arm is placed in the scene, with a Robotiq85 gripper as its end-effector. A $10cm$ cube is placed outside the reachable space of the UR5 robot. A water pipe is grasped by a Robotiq85 gripper and water is flowing out from the pipe. The robot controls the orientation of the pipe to make the cube move to a target area (red area in Fig. 7(b)) with water. It is similar to the pushing task, so we call it water-pushing.

b) Task Implementation: In each step, a 4-d action is needed with the first 3 dimensions representing the delta translation of Robotiq85 along the X, Y, and Z axis and the last 1 dimension representing the gripper width. Note that the initial velocity and volume per unit time of water spray are proportional to gripper width. In each episode, UR5 is reset to a fixed pose and the cube will be sampled within a range. The target area will be sampled within a range around the initial cube position. The observation includes the position, velocity, and width of the gripper and cube, as well as the center of the target area. The reward function is the negative distance between the cube and the center of the target area. The task is regarded as successful when the distance between the cube and the target area center is less than $10cm$. This task is also goal-conditioned, so we use the SAC algorithm with HER.

3) Towel Catching:

a) Task Description: A Kinova Gen2 robot arm is fixed in the scene and a towel will fall down. When the towel is falling down, a wind with random strength and orientation is added to the scene to add some randomization to the movement of the towel. The movement of the end-effector is restricted to a horizontal plane with a fixed height lower than the initial falling height of the towel, so it is required to predict the falling point and catch the towel.

b) Task Implementation: In each step, a 2-d action representing the delta movement along the fixed horizontal plane is needed. In each episode, the Kinova Gen2 robot arm will be reset to a fixed pose but the towel will fall from a random position with a wind of random strength and orientation added to the scene. Since the towel is a high-dimensional deformable object, we calculate the average position among all vertices of the towel. Thus, the observation includes the position and velocity of the end-effector, the average position and velocity of the towel and initial falling position, wind orientation, and strength. The reward function is shown as Equ. 1:

$$reward = 2 - tanh(10 \cdot |v_c^Y|) - tanh(10 \cdot |\mathbf{v_e}|), \quad (1)$$

where v_c^Y is the Y-axis velocity of cloth and $\mathbf{v_e}$ is the 3-d velocity of the end-effector. The task is regarded as successful if the towel is caught at the end of an episode. The task is trained with the SAC algorithm.

D. Butter Pushing: Manipulation experiments with Planning & Control

a) Task Description: By taking the reference video of butter melting, we first mark the trajectory waypoints in RFUniverse. A spatula is attached to the end-effector of a Flexiv robot arm and pushes the butter to follow the trajectory.

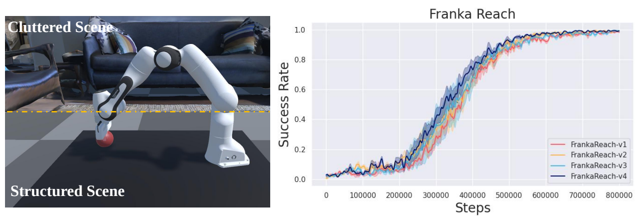
b) Task Implementation: The inverse kinematics calculation and the continuous trajectory are produced by RFMove [49]. To make the butter melt similarly to the reference video, we use 200 particles to simulate the butter from solid to liquid. In the beginning, all particles are solidified to form a butter cube, *i.e.*, all particles are with high viscosity and surface tension value. When particles at the bottom collide with the pot's collider, their viscosity and surface tension value decrease and act like a melted liquid. Let T_i be the temperature of the particle i. To simulate the heat transfer process, the simulation follows [30], and applies $\frac{dT_i}{dt} = -T_i/D_r$ to the particles, where D_r denotes the radiation half time. The value of D_r can be used to suggest the cooling speed of the particles. When bottom particles become liquid, particles with a higher initial position will fall and collide with the pot. Meanwhile, in each step, all particles within a fixed margin will average their viscosity and surface tension. The overall process simulates the heat transfer effect of butter melting from the bottom to the top. All parameters in the experiment are hand-crafted to align with the reference video.

E. Visual pre-trained encoder with Reinforcement Learning

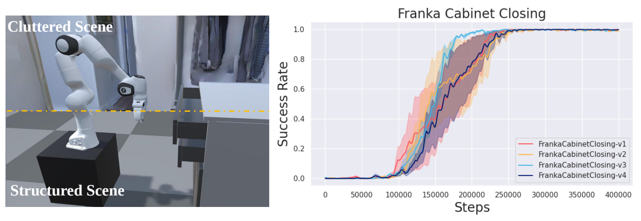
In these experiments, we follow the setting in [47] where we use a pre-trained encoder to encode the image from the current camera perspective. Note that the vector after encoding is the only observation of RL algorithms and the weights of the pre-trained encoder are fixed during training. We build two tasks named Franka Reach and Cabinet Closing and use the identical encoder. For each task, we provide a structured scene version and a cluttered scene version, as well as a camera from a first-person perspective (eye-on-hand) and third-person perspective (eye-on-base) for each version respectively. We use PPO [31] algorithm to train both versions with both perspectives for each task. The experimental results for the relationship between training steps and success rate are shown in Fig. 8.

1) Franka Reach: Franka robot resets to a fixed position $\mathbf{p}_f$ and a target position is randomly sampled from the $30cm \times 30cm \times 30cm$ space around $\mathbf{p}_f$. Franka robot arm needs to let its gripper's grasp point reach the target position. The tolerance is $5cm$ and the target range is a sphere with $5cm$ radius, which is highlighted in the scene. The reward is the negative distance between the grasp point and the target position.

2) Franka Cabinet Closing: A double-layer cabinet with prismatic drawers is initialized with two drawers randomly opened and in a random pose. The Franka robot arm needs to push two drawers back. The reward is the sum of the negative open length of drawers.



(a) Cluttered and structured scene screenshots for Franka Reach task.



(b) Cluttered and structured scene screenshots for Franka Cabinet Closing task.

Fig. 8. Scene screenshots and experimental results for visual pre-trained encoder with reinforcement learning. Cluttered means with the background of the room environment, while structured means without the background. In legend, **v1** means eye-on-hand with structured scene; **v2** means eye-on-base with structured scene; **v3** menas eye-on-hand with cluttered scene; **v4** means eye-on-base with cluttered scene.

V. Conclusion and Future Work

In this paper, we present a novel multiphysics-based simulation environment named **RFUniverse**. With user-friendly interfaces and a Unity-based backend, it provides a chance for the users to seamlessly interact with the virtual world which supports aerodynamics, hydrodynamics, and thermodynamics. We aim at taking the first step towards building a physics realistic world that supports multiphysics coupling effects, with the hope of extending the task scope of current robot manipulation in simulation to the next level.

We would like to propose such a question: Do intelligent agents really need high-precision simulation for embodied AI? As humans, we never compute accurate physics when we are interacting with the real world. Instead, we observe and learn to react. Therefore, we can assume that the world human (or any sensors) observe is also noisy. Seemingly realistic physics simulation could already support many tasks. And for policy learning, such computational inaccuracy could bring more diversity in sensory information and benefit the generalization ability of learned models. Similar ideology is presented in domain randomization techniques.

We regard the RFUniverse simulation environment as an infrastructure for multiphysics-based robot learning and will integrate more physics solvers into the system to keep enlarging the research scope of embodied AI.

REFERENCES

[1] Agx dynamics, Aug 2020. URL https://www.algoryx.se/agx-dynamics/.

[2] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *arXiv preprint arXiv:1707.01495*, 2017.

[3] Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. HoME: a Household Multimodal Environment. In *NIPS 2017's Visually-Grounded Interaction and Language Workshop*, Long Beach, United States, December 2017. URL https://hal.inria.fr/hal-01653037.

[4] David Coleman, Ioan Sucan, Sachin Chitta, and Nikolaus Correll. Reducing the barrier to entry of complex robotic software: a moveit! case study. *arXiv preprint arXiv:1404.3785*, 2014.

[5] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

[6] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, Luca Weihs, Mark Yatskar, and Ali Farhadi. RoboTHOR: An Open Simulation-to-Real Embodied AI Platform. In *CVPR*, 2020.

[7] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022.

[8] Yipai Du, Guanlan Zhang, and Michael Yu Wang. 3d contact point cloud reconstruction from vision-based tactile flow. *IEEE Robotics and Automation Letters*, 2022.

[9] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ManipulaTHOR: A Framework for Visual Object Manipulation. In *CVPR*, 2021.

[10] François Faure, Christian Duriez, Hervé Delingette, Jérémie Allard, Benjamin Gilles, Stéphanie Marchesseau, Hugo Talbot, Hadrien Courtecuisse, Guillaume Bousquet, Igor Peterlik, and Stéphane Cotin. SOFA: A Multi-Model Framework for Interactive Physical Simulation. In Yohan Payan, editor, *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, volume 11 of *Studies in Mechanobiology, Tissue Engineering and Biomaterials*, pages 283–321. Springer, June 2012. doi: 10.1007/8415_2012_125. URL https://hal.inria.fr/hal-00681539.

[11] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, Kuno Kim, Elias Wang, Damian Mrowca, Michael Lingelbach, Aidan Curtis, Kevin Feigelis, Daniel M. Bear, Dan Gutfreund, David Cox, James J. DiCarlo, Josh McDermott, Joshua B. Tenenbaum, and Daniel L. K. Yamins. Threedworld: A platform for interactive multi-modal physical simulation, 2020.

[12] Chuang Gan, Siyuan Zhou, Jeremy Schwartz, Seth Alter, Abhishek Bhandwaldar, Dan Gutfreund, Daniel L. K. Yamins, James J DiCarlo, Josh McDermott, Antonio Torralba, and Joshua B. Tenenbaum. The threedworld transport challenge: A visually guided task-and-motion planning benchmark for physically realistic embodied ai, 2021.

[13] Xiaofeng Gao, Ran Gong, Tianmin Shu, Xu Xie, Shu Wang, and Song-Chun Zhu. Vrkitchen: an interactive 3d virtual environment for task-oriented learning. *arXiv*, abs/1903.05757, 2019.

[14] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[15] Zibra AI Inc. URL https://zibra.ai/.

[16] Arthur Juliani, Vincent-Pierre Berges, Ervin Teng, Andrew Cohen, Jonathan Harper, Chris Elion, Chris Goy, Yuan Gao, Hunter Henry, Marwan Mattar, et al. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018. URL https://github.com/Unity-Technologies/ml-agents.

[17] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*, 2017.

[18] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, et al. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5 (3):3838–3845, 2020.

[19] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael

Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey Kurenkov, C. Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks, 2021.

[20] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *6th Annual Conference on Robot Learning*, 2022.

[21] Jacky Liang, Viktor Makoviychuk, Ankur Handa, Nuttapong Chentanez, Miles Macklin, and Dieter Fox. Gpu-accelerated robotic simulation for distributed reinforcement learning, 2018.

[22] Alexander D. Lindsay, Derek R. Gaston, Cody J. Permann, Jason M. Miller, David Andrš, Andrew E. Slaughter, Fande Kong, Joshua Hansel, Robert W. Carlsen, Casey Icenhour, Logan Harbour, Guillaume L. Giudicelli, Roy H. Stogner, Peter German, Jacob Badger, Sudipta Biswas, Leora Chapuis, Christopher Green, Jason Hales, Tianchen Hu, Wen Jiang, Yeon Sang Jung, Christopher Matthews, Yinbin Miao, April Novak, John W. Peterson, Zachary M. Prince, Andrea Rovinelli, Sebastian Schunert, Daniel Schwen, Benjamin W. Spencer, Swetha Veeraraghavan, Antonio Recuero, Dewen Yushu, Yaqi Wang, Andy Wilkins, and Christopher Wong. 2.0 - MOOSE: Enabling massively parallel multiphysics simulation. *SoftwareX*, 20:101202, 2022. ISSN 2352-7110. doi: https://doi.org/10.1016/j.softx.2022.101202. URL https://www.sciencedirect.com/science/article/pii/S2352711022001200.

[23] Liu Liu, Wenqiang Xu, Haoyuan Fu, Sucheng Qian, Yang Han, and Cewu Lu. Akb-48: A real-world articulated object knowledge base. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[24] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021.

[25] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[26] COMSOL Multiphysics. Introduction to comsol multiphysics®. *COMSOL Multiphysics, Burlington, MA, accessed Feb*, 9:2018, 1998.

[27] NVIDIAGameWorks. Nvidiagameworks/physx: Nvidia physx sdk. URL https://github.com/NVIDIAGameWorks/PhysX.

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[29] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2018.

[30] Bo Ren, Xiao Yan, Tao Yang, Chen-feng Li, Ming C Lin, and Shi-min Hu. Fast sph simulation for gaseous fluids. *The Visual Computer*, 32:523–534, 2016.

[31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[32] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martın-Martın, Linxi Fan, Guanzhi Wang, Shyamal Buch, Claudia D'Arpino, Sanjana Srivastava, Lyne P Tchapmi, Kent Vainio, Li Fei-Fei, and Silvio Savarese. igibson, a simulation environment for interactive tasks in large realistic scenes. *arXiv preprint*, 2020.

[33] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. URL https://arxiv.org/abs/1912.01734.

[34] Edward Smith, Roberto Calandra, Adriana Romero, Georgia Gkioxari, David Meger, Jitendra Malik, and Michal Drozdzal. 3d shape reconstruction from vision and touch. *Advances in Neural Information Processing Systems*, 33:14193–14206, 2020.

[35] Michael Smith. *ABAQUS/Standard User's Manual, Version 6.9*. Dassault Systèmes Simulia Corp, United States, 2009.

[36] Russell Smith. Open dynamics engine, 2008. URL http://www.ode.org/. http://www.ode.org/.

[37] Paloma Sodhi, Michael Kaess, Mustafa Mukadanr, and Stuart Anderson. Patchgraph: In-hand tactile tracking with learned surface normals. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2164–2170. IEEE, 2022.

[38] Virtual Method Studio. URL http://obi.virtualmethodstudio.com/.

[39] Sudharshan Suresh, Zilin Si, Joshua G Mangelson, Wenzhen Yuan, and Michael Kaess. Shapemap 3-d: Efficient shape mapping through dense touch and vision. In *2022*

International Conference on Robotics and Automation (ICRA), pages 7073–7080. IEEE, 2022.

[40] 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. pages 19–49. Springer, 2016.

[41] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012. URL https://ieeexplore.ieee.org/abstract/document/6386109/.

[42] Shaoxiong Wang, Jiajun Wu, Xingyuan Sun, Wenzhen Yuan, William T Freeman, Joshua B Tenenbaum, and Edward H Adelson. 3d shape perception from monocular vision, touch, and shape priors. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1606–1613. IEEE, 2018.

[43] Shaoxiong Wang, Mike Lambeta, Po-Wei Chou, and Roberto Calandra. Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors. *IEEE Robotics and Automation Letters*, 7(2): 3930–3937, 2022.

[44] Fei Xia, Amir R. Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9068–9079, 2018. doi: 10.1109/CVPR.2018.00945.

[45] Fei Xia, William B Shen, Chengshu Li, Priya Kasimbeg, Micael Edmond Tchapmi, Alexander Toshev, Roberto Martín-Martín, and Silvio Savarese. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters*, 5(2):713–720, 2020.

[46] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[47] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[48] Ruolin Ye, Wenqiang Xu, Haoyuan Fu, Rajat Kumar Jenamani, Vy Nguyen, Cewu Lu, Katherine Dimitropoulou, and Tapomayukh Bhattacharjee. Rcareworld: A human-centric simulation world for caregiving robots. *arXiv preprint arXiv:2210.10821*, 2022.

[49] Wenqiang Xu Yongxi Huang, Danqing Li. Rfmove. URL https://github.com/mvig-robotflow/rfmove.

[50] Xiaoshuai Zhang, Rui Chen, Fanbo Xiang, Yuzhe Qin, Jiayuan Gu, Zhan Ling, Minghua Liu, Peiyu Zeng, Songfang Han, Zhiao Huang, Tongzhou Mu, Jing Xu, and Hao Su. Close the visual domain gap by physics-grounded active stereovision depth sensor simulation. *CoRR*, abs/2201.11924, 2022. URL https://arxiv.org/abs/2201.11924.