Robotic Interactive Physics Parameters Estimator (RIPPE)

Atabak Dehban[*1,2], Carlos Cardoso[*1], Pedro Vicente[1],
Alexandre Bernardino[1], and José Santos-Victor[1]

***Abstract*— The ability to reason about natural laws of an environment directly contributes to successful performance in it. In this work, we present *RIPPE*, a framework that allows a robot to leverage existing physics simulators as its knowledge base for learning interactions with in-animate objects. To achieve this, the robot needs to initially interact with its surrounding environment and observe the effects of its behaviours. Relying on the simulator to efficiently solve the partial differential equations describing these physical interactions, the robot infers consistent physical parameters of its surroundings by repeating the same actions in simulation and evaluate how closely they match its real observations. The learning process is performed using Bayesian Optimisation techniques to sample efficiently the parameter space. We assess the utility of these inferred parameters by measuring how well they can explain physical interactions using previously unseen actions and tools.**

I. Introduction

For an agent (artificial or biological) to survive in a physical environment, it must continuously choose actions that derive its sensory states towards goal states. This entails an understanding of the governing laws in the environment, *i.e*. physics.

Developmental studies of humans demonstrate a remarkable ability in young babies to show consistent physical expectations. More specifically, the period between four to six months of age appears to be an important phase in understanding "objectness". *E.g*. infants of 4 months old are more likely to associate two heads of an occluded bar to a single, bounded object compared to younger children [3]. By the age of four months, infants can track a moving ball with their eyes, even if it is occluded by a narrow screen, and by the age of six months, they can reliably track it even when occluded by wider screens [21]. More inline with the present study, Berthier *et al*. [6] showed that nine months old infants reliably predict the reappearance of an occluded ball along a track, only if the track is not blocked by a wall, *i.e*. they give some evidence that babies can predict the interaction between the wall and the ball.

This developmental path suggests that the understanding of objects and their interactions is gained through life experiences continuously as humans engage and interact more and more with their surroundings. The goal of this work is to allow an agent (more specifically, iCub [29]) a similar reasoning capability about *intuitive physics* (see [25] for an overview about the usage of the term).

[1] Institute for Systems and Robotics, Instituto Superior Tećnico, Universidade de Lisboa, Lisbon, Portugal
Email: {adehban,ccardoso,pvicente,alex,jasv}@isr.tecnico.ulisboa.pt
[2] Champalimaud Centre for the Unknown, Lisbon, Portugal
[*] indicates equal contribution.

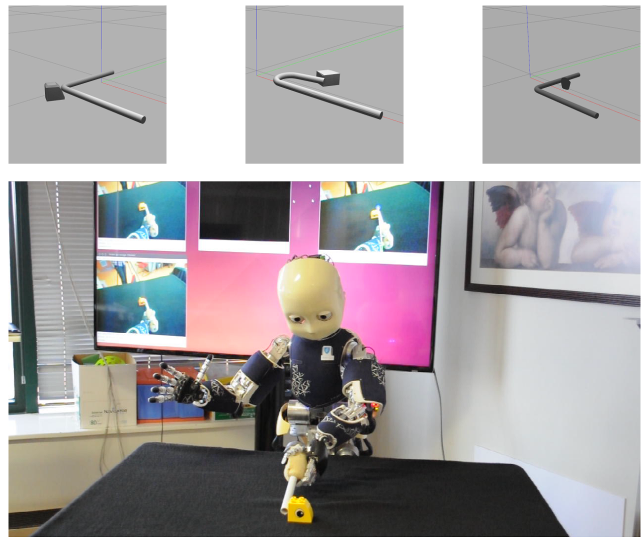


Fig. 1: iCub trying to infer consistent physical parameters of an object by repeating its physical interactions in simulation. These parameters should describe the object behaviour in response to novel tools or actions.

To this end, we allow the robot to interact with the environment and take notes of the effects it causes. However, for artificial agents, there are already many physics simulators available to solve the Newtonian dynamics equations of motion, alleviating the need to learn these dynamics through interaction. To use these simulators, one needs an appearance description of a scene (typically meshes or shape primitives), mass and inertia of the objects and friction coefficients.

In this work, we consider a life-long learner that can interact with objects in an environment via known actions and has access to the 3d shape of those objects. Through these interactions with real world objects, our agent gathers statistics regarding how objects behave with respect to known actions. Later, the robot repeats similar interactions with the 3D models of objects in simulation and adjusts the physical parameters of those objects (*e.g*. mass, damping and friction coefficients, *etc*. assuming uniform mass distribution) such that the simulated observations get closer and closer to real observations.

When faced with a new action or tool, the robot bootstraps these physical parameters based on the previous interactions it had with that objects and successfully predicts how would the object behave as a result of the new interaction.

This paper has the following contributions:

1) We infer consistent physical parameters using only snapshots of the environment before and after physical interactions by integrating a physics simulator.
2) Our method is applicable to data collected by a robot interacting with complex objects and tools.

Our code is accessible in the following repository [1].

The rest of the paper is organised as follows: in section II we present a brief overview over the lines of research that are related to our work. We continue in section III to describe in more details how we plan to tackle our research question. Section IV describes our thorough tests and evaluations, showcasing the applicability of our solution when the agent is asked to reason about interaction outcomes from novel tools and actions. Finally, we conclude the article and propose promising future research directions in section V.

II. Related Work

A. Understanding Physics from Vision

Accelerated by the growing popularity of deep neural networks and their unreasonable performance in many computer vision tasks, there have been a growing interest in bridging visual perception to physics [30], [35], [36], [5], [10], [14]. In addition, reconstructing an observed scene in simulation opens many opportunities for robotic tasks, *e.g.* [28] used 3D point clouds to reconstruct a shape in simulation and attempt several grasps. This work was only concerned with shapes and neglected other physical properties.

Abelha and Guerin [2] created mesh representations of objects in Gazebo [23] and by holding the tool in different ways, determined whether it can be used to achieve a particular task on a particular object. They used super-quadrics and super-paraboloids to define a feature vector representing the geometry of the tool together with how it is being held, and by augmenting this information with the tools weight, they managed to generalise the simulation experiments over various tool without re-running the simulation. From these experiments, it is not clear how to generalise the learned knowledge to different tasks and situations.

Battaglia *et al.* [5] and Chang *et al.* [10] learn physical laws and parameters from observing videos. They rely on an external tracker to detect the objects and measure some of their states. The interaction space in these works has an “addition property” where effects of interaction of multiple objects together can be summed. As mentioned in the introduction, we believe learning the physics together with the parameters is redundant for artificial agents when more accurate physics engine are available (in fact, these methods rely on physics engines to provide supervised training data).

Wu *et al.* [35] learns physical parameters such as mass, spring stiffness, *etc.* from observing videos and assuming knowledge of underlying physical equations. Their method achieves remarkable accuracy, but they are limited to the range of physics equations under consideration. Our framework is more general, and can trivially incorporate the same inferred parameters, as long as the selected physics engine allows simulation of these scenarios.

Scene UNderstanding (SUN) RGB-D was used by [30] to predict direction of motion of objects in simulation if a 3D force was applied to a specific pixel. Since aligned Computer-Aided Design (CAD) models do not exist in SUN-RGB-D, they reconstructed objects in the simulation as cubes. This simplification in terms of shape, exacerbated by the fact that they had no ground to real physical interactions, were among the reasons that the model sometimes made extraordinary mistakes.

Mar *et al.* [26] used 2D and 3D features extracted from the shape of a tool to allow an iCub robot successfully employ a novel tool to achieve a desired task, however, the robot needs re-training before it can apply its knowledge for new actions.

Another line of research that studies the effects of actions on objects relies on the concept of affordances [16] to learn a joint distribution over the representations of objects, tools, actions, and effects. Jamone *et al.* [20] overviews the applications and theories of affordances in various fields such as robotics and neuroscience. For example, [31] uses the knowledge of affordances to generalise the outcome of interaction experiments from robot hands to unseen tools. Previously, we proposed a Variational Auto Encoder (VAE) [22] to learn a joint distribution over features of tools, objects, actions and effects [13]. Both of the last two works only show generalisation to new tools and objects, while generalisation to new actions is out of their scope.

Galileo [36] is maybe the work which is most similar to our formulation as it also leverages a physics engine. However, there are several important aspects which differentiate the present article from this work:

- Galileo [36] is only concerned with simple parametric shapes whereas we use real scanned models of objects and reconstructed models of tools.
- We use physical robotic interactions with the environment, in contrast with [36] where they rely on passively recorded videos.
- Due to the availability of clean videos, [36] uses a tracker to obtain an informative time-series of observation states. However, in robotic scenarios, such a rich source of information is rarely available, for example as a result of self occlusion. Thus, we have reformulated the problem to use the information in snap-shots of environment, before and after physical interactions.

Since the tracker and code of [36] is not publicly available, we cannot directly compare our method with Galileo. To foster research in learning generic physical interactions, the code of our experiments are publicly available [1].

B. Available Environments and Datasets

1) Interactive Datasets: In general, creating datasets for interactive tasks is very time-consuming, thus, a possible approach is to couple the collected data with a physics engine to create an interactive framework such as Gibson Environment [37] or AI2-THOR [24]. AI2-THOR is a 3d modelled synthetic home environment where an agent can interact with objects and perform actions such as opening a fridge or putting a towel on a rack. Gibson Environment, on the other hand, has 211 km^2 of 3D scanned real indoor spaces. Both frameworks focus more on tasks like navigation and have limitations for our purposes, *i.e*. AI2-THOR is not based on real data, and Gibson Environments does not have movable objects to interact.

2) Photo Realistic Environments: Photo Realistic environments can be used to enhance the fidelity of rendered images from existing simulators, to reduce the infamous reality gap*. On one hand, there is the Habitat-Sim [32] developed by Facebook Research with built-in support for several datasets (including Gibson) which can achieve several *frames per second*. Additionally, unrealROX [27] builds on top of the known game engine: Unreal, using its physics engine and rendering a robotic platform on a realistic simulation environment. Both environments can be used to generate a realistic large-scale datasets, however they have no real interaction, therefore they cannot be used to infer physical properties of real objects.

3) Physics 101: The dataset used in [35], [36] is composed of 10,000 video clips containing 101 objects of various materials and appearances in different physical experimental settings. Even though these videos can be used to infer several physical properties, such un-occluded clean views of the environment over fixed backgrounds are orthogonal to robotic settings with self occlusions and moving cameras.

4) Affordances dataset: The Affordances dataset [12] contains information about the effect of performing different actions on objects. The data consists of stereo image pairs corresponding to more than 1320 robot trials. The trials include 4 actions, push, pull, push-right and push-left performed by iCub with 3 tools on 11 objects from the Yale-CMU-Berkeley Object and Model set (YCB) [9] and each action was repeated at least 10 times.

Since the 3d models of objects in YCB are publicly available, we decided to use a subset of this dataset to infer consistent physical parameters and evaluate them against hold out actions and tools.

III. Methods

A. Overview of the pipeline

In this work, we are concerned with estimating physical parameters from physical interactions. However, there are several challenges to achieve this goal which we will describe and propose our solutions.

First, the observations may not contain enough information to recover the true value of a parameter. For example, in our scenario, the robot pushes the objects on a table with various tools and measures the displacement of the object as observation data. Theoretically, a lighter object with a higher friction coefficient can travel the same distance as a heavier object with a smaller friction coefficient. Thus, in order to assess the utility of these parameters, we assume that if a set of parameters can improve the similarity between simulated and real displacements over some actions and tools, they should consistently improve the similarity between simulated and real effects over unseen tools and actions. This assumption is validated by our experiments.

Second, the trajectory of an object's motion in response to an executed action depends on the exact relative initial position and speed of execution, which are both unknown in most realistic robot scenarios. In our case, where objects move on a table along x and y axis, we assume the effect of applying a given action with a given tool has a 2D Gaussian distribution with full co-variance. Thus, our cost function would be the average of Kullback-Leibler (KL) divergences of simulated and observed displacements over different actions and tools(eq. (1)).

Third, to avoid using specific physical relations for each scenario, we rely on a physics engine to simulate the environment. However, sampling from a physics simulator is costly, especially as we need multiple samples for each experiment and parameter configuration to obtain data points that are reliable. *I.e*. we have an optimisation problem where the cost function is non-differentiable (depending on samples from a physics simulator) and cost evaluations are time-consuming. To tackle both of these challenges, we propose to use Bayesian Optimisation (BO) which does not require the gradient of the cost function and is known to be sample efficient.

B. Bayesian Optimisation

Given the 3D object models, for a robot to meaningfully use a physics simulator to predict the effects of interactions, it still needs to estimate simulation parameters such that simulated interactions optimally match the real ones. However, it is very complicated to analytically estimate the simulation outcomes, let alone their similarity to real observations (our measure of similarity is defined in eq. (1)). Thus, instead of directly optimising this function, we will optimise a probabilistic surrogate of the similarity.

Formally, the problem is based on finding the optimum (minimum) of an unknown real valued function $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} \subset \mathbb{R}^d$ is a compact space, and $d \geq 1$ reflects the number of the physical parameters to estimate, with a maximum budget of N evaluations of the target function f. In our scenario, $f(\mathbf{x})$ is the continuous KL divergence (D_{KL}) between the real and simulated distributions of effects of different actions on the object whose parameters are the be estimated. The distributions of the effects of real actions are acquired empirically from the dataset of robot interactions (Affordance dataset) and are kept fixed. One the other hand, the distributions of the effects of simulated actions depend on the physical parameters that we want to optimise such that KL divergence between the real and the simulated action effects are as similar as possible, so we aim at minimising

*the discrepancy between the real and the simulated, computer generated environments is often referred to as the reality gap [7].

$$
\begin{aligned}
f(\mathbf{x}) &= D_{\mathrm{KL}}(R\|S(\mathbf{x})) = \int [\log(r) - \log(s(\mathbf{x}))] r dx \\
&= \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_{S(\mathbf{x})}|}{|\boldsymbol{\Sigma}_R|} - m + \operatorname{tr}\left\{\boldsymbol{\Sigma}_{S(\mathbf{x})}^{-1}\boldsymbol{\Sigma}_R\right\}\right. \\
&\quad \left. + \left(\boldsymbol{\mu}_{S(\mathbf{x})} - \boldsymbol{\mu}_R\right)^T \boldsymbol{\Sigma}_{S(\mathbf{x})}^{-1}\left(\boldsymbol{\mu}_{S(\mathbf{x})} - \boldsymbol{\mu}_R\right)\right],
\end{aligned} \tag{1}
$$

where R and $S(\mathbf{x})$ are multivariate normal distributions with mean $\boldsymbol{\mu}$ and co-variance matrix $\boldsymbol{\Sigma}$ and m is the number of components and in this paper we have $m = 2$ since we define observations/effects as 2D displacements of the object on a table. R are the observed effects on the real world, and $S(\mathbf{x})$ the simulated ones that depend on our parameters $\mathbf{x} \in \mathcal{X}$.

Figure 2 shows that when the simulated samples (orange dots) are very different from the observed real samples (blue crosses), the cost function is high (fig. 2a) and *vice versa* (fig. 2b).

Our objective is to minimise eq. (1), however, estimating S over the whole state–space is in-tractable and we can only evaluate eq. (1) at a number of points.

The BO consists of two stages. First, given a query point $\mathbf{x}_i$ and outcome $y_i = f(\mathbf{x}_i) + \varepsilon$, ε representing the measurement noise, we update a probabilistic surrogate model of f, a distribution over the family of functions $P(f)$, where the target function f belongs. Gaussian Process (GP) is a popular choice for this family which can be built incrementally by sampling over the input–space [8]. More specifically, given a dataset $\mathcal{D}_t = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_t, y_t)\}$ of evaluations, where t is the number of samples, y_t is the evaluation of eq. (1) at $\mathbf{x}_t$, and a query point $\mathbf{x}_{t+1}$, the surrogate for the target has the distribution $\hat{y}_{t+1} \sim \mathcal{N}(\mu_{t+1}, \sigma^2_{t+1})$ where:

$$
\mu_{t+1} = \mathbf{k}^T\mathbf{K}^{-1}\mathbf{y}_{1:t}\ ; \quad \sigma^2_{t+1} = 1 - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k}, \tag{2}
$$

and we have:

$$
\begin{aligned}
\mathbf{K} &= \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & \ldots & k(\mathbf{x}_1,\mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t,\mathbf{x}_1) & \ldots & k(\mathbf{x}_t,\mathbf{x}_t) \end{bmatrix}, \\
\mathbf{k} &= \begin{bmatrix} k(\mathbf{x}_{t+1},\mathbf{x}_1) & k(\mathbf{x}_{t+1},\mathbf{x}_2) & \cdots & k(\mathbf{x}_{t+1},\mathbf{x}_t) \end{bmatrix}
\end{aligned} \tag{3}
$$

where $k : \mathcal{X} \times \mathcal{X} \to (0, 1]$ is a bounded measure of proximity between two points. One choice of this function is the Matérn kernel [34, p. 84]:

$$
k(\mathbf{x}_i,\mathbf{x}_j) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\,\|\mathbf{x}_i - \mathbf{x}_j\|}{l}\right)^{\nu} H_\nu\left(\frac{\sqrt{2\nu}\,\|\mathbf{x}_i - \mathbf{x}_j\|}{l}\right),
$$

such that l is a hyper-parameter, dynamically determined per feature to maximise the log-likelihood of the observations $\mathcal{D}_t$, $\|\cdot\|$ is the L2 norm, $\Gamma(\cdot)$ is the Gamma function, and $H_\nu(\cdot)$ is the Bessel function of order ν. We have kept $\nu = 2.5$ during our experiments without any tuning.

Second, a Bayesian decision process, where an acquisition function $A : \mathcal{X} \to \mathbb{R}^+$ uses the information gathered in the GP to decide on the next best point $\mathbf{x}_{t+1} = \arg\max_{\mathbf{x}} A_t(\mathbf{x})$ to query/sample. The goal is to guide the search to the optimum, while balancing the *exploration vs. exploitation* trade-off, given the consideration that sampling from eq. (1) to augment $\mathcal{D}$ is time–consuming.

Various acquisition functions are proposed in the literature [15] and sometimes, it is not clear which one better suits a particular problem. Thus, we propose to apply the GP Hedge strategy [19] which uses three different acquisition functions: Expected Improvement (EI), negative Probability of Improvement (PI) and Lower Confidence Bound (LCB). This strategy calculates the next query point according to each criterion and chooses one to use in the next iteration according to the softmax$(\eta \cdot g_t)$ probability, where the hyper-parameter η is initialised with a positive real value and g_t is the gain updated in each iteration according to the expected value of the new GP.

These acquisition functions are based on maximising the expected value of a utility function. One simple utility function can be defined over $\mathcal{X}$ as:

$$
\mathrm{u}^{\mathrm{PI}}(\mathbf{x}) = \begin{cases} 0 & \hat{y}(\mathbf{x}) > \mathrm{y}_n^{bopt} \\ 1 & \hat{y}(\mathbf{x}) \le \mathrm{y}_n^{bopt} \end{cases}, \tag{4}
$$

where $\mathrm{y}_n^{bopt} = \min(y_{1:t})$, *i.e.* the best outcome found until now (iteration t). Then we have $A^{\mathrm{PI}} = \Phi(z)$, $\Phi(\cdot)$ corresponds to the Cumulative Density Function (CDF) of the standard normal distribution (zero mean, unit variance) and $z = (\mathrm{y}_n^{bopt} - \mu(\mathbf{x}))/\sigma(\mathbf{x})$ [33]. Also, the pair $(\mu(\mathbf{x}), \sigma^2(\mathbf{x}))$ are the predictions computed in eq. (2).

The utility function corresponding to EI is $\mathrm{u}^{\mathrm{EI}}(\mathbf{x}) = \max(0, \mathrm{y}_n^{bopt} - \hat{y}(\mathbf{x}))$, resulting in the acquisition function:

$$
\begin{aligned}
&A^{\mathrm{EI}}(\mathbf{x}) = \mathbb{E}_{(\hat{y}|\mathcal{D}_t)}[\max(0, \mathrm{y}_n^{bopt} - \hat{y}(\mathbf{x}))] = \\
&\begin{cases} \left(\mathrm{y}_n^{bopt} - \mu(\mathbf{x})\right)\Phi(\gamma) + \sigma(\mathbf{x})\phi(z) & \text{if } \sigma(\mathbf{x}) > 0 \\ 0 & \text{if } \sigma(\mathbf{x}) = 0 \end{cases}
\end{aligned}
$$

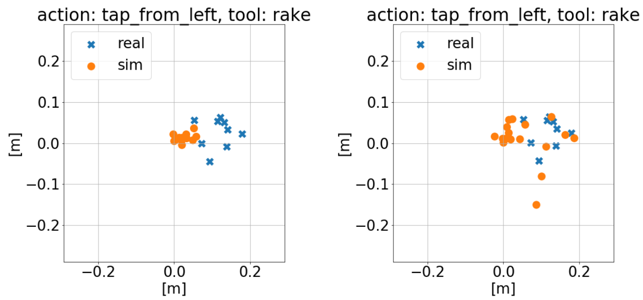
where $\phi(\cdot) = \mathcal{N}(\cdot; 0, 1)$.

Finally, regarding LCB, this acquisition function is defined as $A^{\mathrm{LCB}} = -\mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$, where $\beta > 0$ is a parameter to trade-off between exploitation ($\mu(\mathbf{x})$) and exploration ($\sigma(\mathbf{x})$) [17, p. 33].

C. Using simulation to extract physical parameters

Given a set of parameters selected by the Bayesian optimiser, we need to evaluate how close does the simulation outcomes resemble the real experiment. We re-created the setup of the Affordances dataset [12] in the Bullet physics engine [11] and assume as known variables: i) the shape and trajectory of the robot and tool, and ii) the shape and starting position (but not orientation) of the objects.

Using the object's 3D shape, we compute its convex hull, which is necessary for the physics engine to efficiently calculate stable contact forces between the meshes. From the obtained convex shape, we assume that the object has a uniform density and algorithmically compute its inertia matrix and centre of gravity. The lateral and rolling frictions however cannot be computed in this way and are found through the Bayesian optimisation. These friction parameters



(a) KL divergence 122.7 (b) KL divergence 0.96

Fig. 2: Samples explored in the simulation for one object (yball). The axes are modified to reflect robot's viewpoint, *e.g.* by tapping from left, objects generally move to the right side. In (a) the orange dots drawn from the simulation show a very different distribution than the blue crosses resulting in a large KL divergence. This is due to the un-matched physical parameters. In (b) we can see the opposite with parameters found as the optimisation converges.

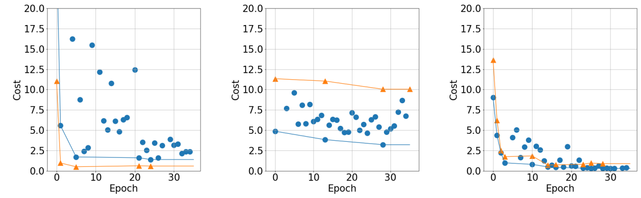
would require complex experiments for the robot to measure directly and in addition, even if the parameters can be accurately measured, it is usual to adjust these parameters by trial and error in order to stabilise the simulation. For each combination of parameters (two friction coefficients and mass, $\mathbf{x}_t, d = 3$), object, tool, and action we perform the action in simulation 500 times by randomising the object's initial orientation. This randomisation is necessary because we do not have access to the ground-truth orientations of the object, however, this step introduces considerable noise into our evaluation of eq. (1), justifying the need to perform the simulation at least 500 times. The outputs of the simulations are the final (x,y) positions of the object after each repetition. Finally, the 500 position samples are used to fit a Gaussian distribution S empirically and the KL divergence to the observed distribution R is computed. This divergence is returned as a cost y_t to guide the optimiser.

IV. Experiments and Results

In all the following experiments, we have used Bullet physics engine [11] to simulate interactions with the virtual world and scikit optimisation package [18].

As explained before, if the robot wants to use the simulator as a reliable source of information about interaction outcomes in the real world, it needs to provide adequate simulation parameters that specify the physical properties of the target object, the inertia matrix, centre of gravity, friction coefficients, the weight of the object, *etc*.

However, our robot has no way of accurately knowing or directly measuring these parameters so it must estimate them in an interactive manner by examining the simulation outcomes through trial and error guided by the KL divergence cost function described in the previous section. In order to accurately estimate the KL divergence from the simulated experiments we repeat the simulation 500 times for each set



(a) YCB yball. (b) YCB ylego. (c) YCB lemon.

Fig. 3: Blue dots are the cost evaluations for each set of parameters while searching the simulations parameter space with two actions (push and tap from left) and with a tool (rake). The parameters are the lateral friction, the rolling friction, and mass. The orange triangular markers are the cost when using the corresponding parameters for an un-experienced action (tap from right with hook). The parameters that improved the cost in training are connected with a solid line. Abbreviations in [12].

of (object, parameters, tool and action). The lateral friction is bounded to values in the range $[10^{-2}, 5]$, the rolling friction is in the range $[10^{-12}, 10^{-3}]$, and the mass varies between $[0, 200]$ grams.

To see if the learnt parameters can generalise to new interactions, namely unseen tool and un-tested action, we selected three objects from the Affordances dataset [12] and estimated physical parameters using the interaction data from two actions (push and tap from left) and a tool (rake). Due to the *exploration–exploitation* trade-off of BO, only some of the parameters during the optimisation course result in improving/lowering the cost. At test time, we only evaluated the parameters that resulted in an improvement on the train set. In figure 3, the decreasing costs during optimisation and the corresponding test-set evaluations are connected with solid lines. According to our results, improving the performance in the train set generally improves the performance on the un-experienced novel actions *i.e.* when there is a decrease in the train set, the cost on the test set is also decreased.

V. Conclusions and Future Work

In this document, we have presented a complete framework to infer consistent physical parameters which allow a robot to predict the effects of applying novel actions to objects. Our experiments demonstrate that even with very noisy and sparse observations from real robot interactions, it is possible to extract physical parameters that can explain object's motion in response to unseen interactions.

In our current formulation, we are only interested in generalising the learnt knowledge to unseen tools and actions. However, for a new object, there is the need for new interactions before we can predict the effects of novel actions, *i.e.* the knowledge of past interactions with different objects cannot be leveraged. Learning a representation of an object related to its intrinsics [4] can allow an agent to use the interactions with objects that have similar intrinsics to bootstrap its estimate of physical parameters that determine the outcome of interactions.

Finally, our proposed solution for estimating physical parameters is generic, but in this work we have only show cased it on a single scenario of a robot interacting with objects on a table using different tools. In the future, we plan to extend these experiments in more general settings and incorporate other sources of information during interaction, *e.g.* tactile sensors.

ACKNOWLEDGEMENTS

This work is partially supported by the Portuguese Foundation for Science and Technology (FCT) project [UID/EEA/50009/2019] and [PD/BD/135115/2017].

REFERENCES

[1] Robotic Interactive Physics Parameters Estimator (RIPPE) code. `https://github.com/carlos-cardoso/RIPPE`, 2019.

[2] P. Abelha and F. Guerin. Learning how a tool affords by simulating 3d models from the web. In *IEEE–RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4923–4929, 2017.

[3] D. Amso and S. P. Johnson. Learning by selection: Visual search and object perception in young infants. *Developmental psychology*, 42(6):1236, 2006.

[4] H. Barrow, J. Tenenbaum, A. Hanson, and E. Riseman. Recovering intrinsic scene characteristics. *Computer Vision Systems*, 2:3–26, 1978.

[5] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4502–4510, 2016.

[6] N. E. Berthier, B. I. Bertenthal, J. D. Seaks, M. R. Sylvia, R. L. Johnson, and R. K. Clifton. Using object knowledge in visual tracking and reaching. *Infancy*, 2(2):257–284, 2001.

[7] J. Borrego, R. Figueiredo, A. Dehban, P. Moreno, A. Bernardino, and J. Santos-Victor. A generic visual perception domain randomisation framework for gazebo. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 237–242, 2018.

[8] E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[9] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar. Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. *IEEE Robotics Automation Magazine*, 22(3):36–52, Sept 2015.

[10] M. B. Chang, T. Ullman, A. Torralba, and J. B. Tenenbaum. A compositional object-based approach to learning physical dynamics. In *International Conference on Learning Representations (ICLR)*, 2017.

[11] E. Coumans. Bullet physics engine. Open Source Software: `http://bulletphysics.org`, 2010.

[12] A. Dehban, L. Jamone, A. R. Kampff, and J. Santos-Victor. A moderately large size dataset to learn visual affordances of objects and tools using icub humanoid robot. In *ECCV Workshop on Action and Anticipation for Visual Learning*, 2016.

[13] A. Dehban, L. Jamone, A. R. Kampff, and J. Santos-Victor. A deep probabilistic framework for heterogeneous self-supervised learning of affordances. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 476–483, 2017.

[14] K. Fragkiadaki, P. Agrawal, S. Levine, and J. Malik. Learning visual predictive models of physics for playing billiards. *arXiv preprint arXiv:1511.07404*, 2015.

[15] R. Garnett. Bayesian methods in machine learning (course notes for CSE 515T). `URL: https://www.cse.wustl.edu/~garnett/cse515t/spring_2019/files/lecture_notes/12.pdf`, March 2019. [Online; accessed June 28, 2019].

[16] J. J. Gibson. *The Ecological Approach to Visual Perception*. Boston, MA: Houghton Mifflin, 1979.

[17] J. González. Gaussian process masterclass. `URL: http://gpss.cc/gpmc17/slides/LancasterMasterclass_1.pdf`, February 7 2017. [Online; accessed June 28, 2019].

[18] T. Head, MechCoder, G. Louppe, I. Shcherbatyi, fcharras, Z. Vincius, cmmalone, C. Schrder, nel215, N. Campos, T. Young, S. Cereda, T. Fan, rene rex, K. K. Shi, J. Schwabedal, carlosdanielcsantos, Hvass-Labs, M. Pak, SoManyUsernamesTaken, F. Callaway, L. Estve, L. Besson, M. Cherti, K. Pfannschmidt, F. Linzberger, C. Cauet, A. Gut, A. Mueller, and A. Fabisch. scikit-optimize/scikit-optimize: v0.5.2, Mar. 2018.

[19] M. Hoffman, E. Brochu, and N. de Freitas. Portfolio allocation for bayesian optimization. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pages 327–336, 2011.

[20] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor. Affordances in psychology, neuroscience and robotics: a survey. *IEEE Transactions on Cognitive and Developmental Systems*, 2016.

[21] S. P. Johnson, D. Amso, and J. A. Slemmer. Development of object concepts in infancy: Evidence for early learning in an eye-tracking paradigm. *Proceedings of the National Academy of Sciences*, 100(18):10568–10573, 2003.

[22] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[23] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IEEE–RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2149–2154. IEEE, 2004.

[24] E. Kolve, R. Mottaghi, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi. Ai2-thor: An interactive 3d environment for visual ai, 2017.

[25] J. R. Kubricht, K. J. Holyoak, and H. Lu. Intuitive physics: Current research and controversies. *Trends in cognitive sciences*, 21(10):749–759, 2017.

[26] T. Mar, L. Natale, and V. Tikhanoff. A framework for fast, autonomous and reliable tool incorporation on icub. *Frontiers in Robotics and AI*, 5:98, 2018.

[27] P. Martinez-Gonzalez, S. Oprea, A. Garcia-Garcia, A. Jover-Alvarez, S. Orts-Escolano, and J. G. Rodríguez. Unrealrox: An extremely photorealistic virtual reality environment for robotics simulations and synthetic data generation. *arXiv preprint arXiv:*, abs/1810.06936, 2018.

[28] Z.-C. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz. General 3d modelling of novel objects from a single view. In *IEEE–RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3700–3705. IEEE, 2010.

[29] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, and L. Montesano. The iCub humanoid robot: An open-systems platform for research in cognitive development. *Neural Networks*, 23(8):1125–1134, 2010.

[30] R. Mottaghi, M. Rastegari, A. Gupta, and A. Farhadi. what happens if... learning to predict the effect of forces in images. In *European Conference on Computer Vision (ECCV)*, pages 269–285. Springer, 2016.

[31] G. Saponaro, P. Vicente, A. Dehban, L. Jamone, A. Bernardino, and J. Santos-Victor. Learning at the ends: From hand to tool affordances in humanoid robots. In *2017 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pages 331–337, Sep. 2017.

[32] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. Habitat: A platform for embodied ai research. *arXiv preprint arXiv:*, 2019.

[33] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2951–2959, 2012.

[34] C. K. Williams and C. E. Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT Press Cambridge, MA, 2006.

[35] J. Wu, J. J. Lim, H. Zhang, J. B. Tenenbaum, and W. T. Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *British Machine Vision Conference (BMVC)*, volume 2, page 7, 2016.

[36] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 127–135, 2015.

[37] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese. Gibson env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018.