SENSORIMOTOR GRAPH: Action-Conditioned Relational Forward Model Learning of Robotic Soft Hand Dynamics

João Damião Almeida* [1], Paul Schydlo* [1,2], Atabak Dehban[1] and José Santos-Victor[1]

***Abstract*— The human hand is a complex system, with many interacting parts, that generates a rich motor repertoire with complex control dynamics. Human infants acquire such sophisticated motor skills through sensorimotor learning and development, by observing the hand/fingers movements in response to the activation of specific muscles, known as motor babbling. In this work, we take inspirations from sensorimotor learning, and apply a Graph Neural Network to the problem of modelling a non-rigid kinematic chain such as a robotic soft hand, taking advantage of two key properties: 1) The system is compositional, that is, it is composed of simple interacting parts, 2) it is order invariant, *i.e.* only the structure of the system is relevant for predicting future trajectories. We denote our model as the "Sensorimotor Graph". We benchmark the "Sensorimotor Graph" against non-structured baselines and evaluate the model robustness to gradually more adverse conditions. We find that our model performs on par with studied baselines in very basic scenarios while significantly outperform them in more challenging tasks such as configuration changes and varying number of fingers/joints.**

I. Introduction

A. Motivation

During the first few months of life, newborns go through a phase of intensive sensorimotor learning, performing endless "experiments" with their own bodies, using motor babbling and observing the motor outcomes, to learn how to control their arms and hands [1]. This learning process to understand the structure and behaviour of their own hands, is also observed when, for example, an amputee has to learn how to adapt to a prosthesis after a replacement surgery.

Consider now an analogous situation for a robotic manipulator, where a robot needs to understand how to use its end-effector. To effectively control this system we first need to write down the equations which govern the system. This approach is feasible when the relations between the different parts are simple. This is not the case for systems with complex non-trivial dynamics such as soft robotics, fishing nets, cloth, or rubbery materials commonly used in prosthetic limbs. In this work we seek to explore a self-calibration strategy inspired by infant sensorimotor learning through motor babbling: by observing the effects of actuation signals on a robotic soft-hand we want to infer the underlying structure and dynamics in the form of a Sensorimotor Graph.

B. Related Work

*1) **Forward Models**:* In classical approaches, the system model and parameters are assumed to be known in advance,

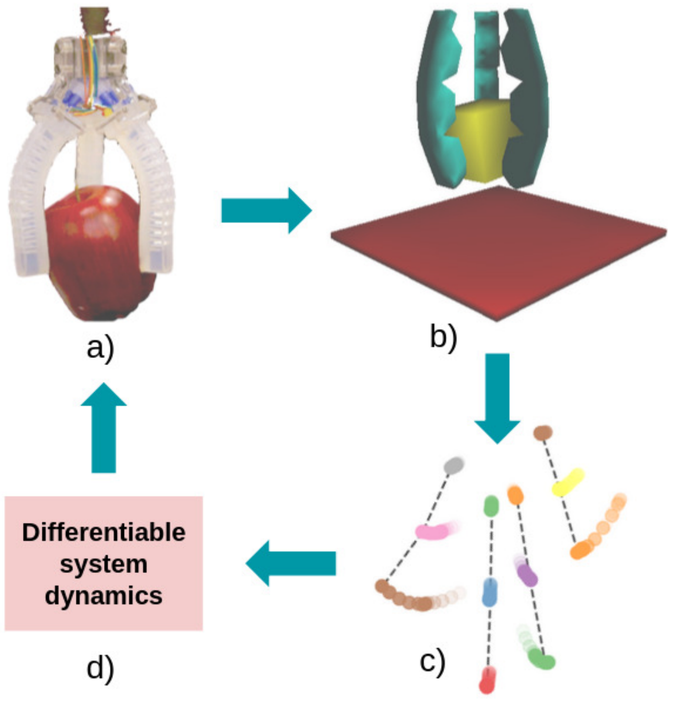

Fig. 1. a) Robotic Soft hand system [2] b) Soft material simulation in SOFA [3] c) Modelling part interaction with graph structured model d) Differentiable system dynamics can be used in an optimal control formulation to control the system [4].

and the controller design makes use of this information [5]. However, when the complexity of the system increases, considering the kinematics, under-actuation, and dynamics of dexterous hands, it becomes very difficult to explicitly write precise and detailed model equations, thus hampering the model-based control system design. Adaptive control emerged as a partial solution to this problem [6]. By allowing parameters of the model to be estimated, it adapts to some levels of uncertainty regarding model parameters, such as e.g.: weight of objects. However, this approach is limited to a fixed structure of the model, which can limit its application to complex systems.

This has lead the community to consider learning based approaches to tackle the effects of under-modeling [7], [8]. Reinforcement learning [9] is one such method which learns to model and control a system by learning to map a system state to a control signal, trained by optimizing a reward signal. The system learns to control implicit system dynamics. Notwithstanding, reinforcement learning assumes the availability of interaction samples and a reward signal,

*Equal contribution [1]Institute for Systems and Robotics, Instituto Superior Técnico, University of Lisbon [2]Machine Learning Department, Carnegie Mellon University, USA

that can be costly to collect.

Model based control, on the other hand, decouples learning of the system dynamics and control of the system, reducing the need for costly interaction samples. Optimal control formulations based on differential system dynamic model [4], [10] optimize a cost function conditioned on the system dynamics, reference state and actuation signals.

Learning based approaches are generally based on two kinds of representations, in structured space, e.g.: 2D or 3D position, or directly on the image space. Image space representations take as input a sequence of frames and predict future frames, modelling the dynamics of the environment in pixel space. This kind of model acts either by encoding the sequence into a compressed representation of the information contained in the frames, the so called latent space [8], and then learning dynamics over this space, or directly warping past pixels into the future. This representation is not invariant to distribution shifts such as lighting or camera pose changes.

Methods based on explicit 3D representations on the other hand are lighting and camera pose invariant. These approaches are based on either manually annotating the data, adding additional instrumentation in the form of markers, or learning to extract and identify key-point representations in an unsupervised manner [11], [12], [10]. More structured representations have gained attention with the recent growing adoption of graph neural networks, which enable learning based methods to exploit the structure inherent in the problem domain [10], [13], see next section.

We focus on the question of how to best model a non-rigid kinematic chain, that is, a dynamical system of interconnected parts. We will assume explicit 3D point position representations, which allow us to limit the scope of this work to the problem of modelling the non-rigid kinematic chain of a robotic soft hand. Additionally, this representation is invariant to lighting and camera pose changes. In a real system the points can be obtained by either placing markers on the different fingers or by learning to segment a RGB-D camera stream into different finger segments.

*2) **Graph Neural Networks:*** Neural networks have enabled great progress in different areas of robotics and computer vision, yet they are very sensitive to distribution shifts such as camera pose or lighting changes. Recently, graph neural network models attempt to bridge the divide between classical structured and more recent deep learning data-driven approaches [12]. The combination mitigates shortcomings by reasoning directly over explicit entities and their relations, the so called relational inductive bias. Imposing architectural constraints, this class of methods captures the inherent compositionality of the system, reasoning about the environment in terms of entities and relations. This is postulated to be similar to how humans reason about their environment [12].

Graph neural networks are structured around a set of entities or nodes/vertices, v_i, and relations or edges between them, e_k. Together the nodes and edges form a graph that defines the dependencies among variables.

Information is propagated through a message passing mechanism in the form of successive edge and node update [14]. Each edge is updated with the function ϕ^e accumulating and conditioning on the neighbouring nodes. Each node is updated with a function ϕ^v accumulating and conditioning on the surrounding edges. Successive iterations of these two functions converge towards the final node values. This mechanism is similar to the way information is collected in a convolutional kernel, where a pixel value is updated as a function of the information of the neighbouring pixels. Following the notation in [12] this can be written as:

$$\begin{aligned} &\mathbf{e}'_k = \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}\right) \quad \overline{\mathbf{e}}'_i = \rho^{e\rightarrow v}\left(E'_i\right) \\ &\mathbf{v}'_i = \phi^v\left(\overline{\mathbf{e}}'_i, \mathbf{v}_i\right) \end{aligned} \tag{1}$$

where r_k is the receiving node and s_k the sending node. $\mathbf{e}'_k$ and $\mathbf{v}'_i$ denote the new edge and node values respectively. The messages are grouped in the following manner.

$$E'_i = \left\{\left(\mathbf{e}'_k, r_k, s_k\right)\right\}_{r_k=i, k=1;N^e} \tag{2}$$

Since the accumulation function, ρ, is order invariant, this kind of network is especially robust to changes in the order of the points, as long as the topology remains invariant. We take advantage of this property by exploiting the order invariance to make the model robust to possible re-identification or tracking issues.

This structure is specially interesting for physical systems with structural constraints and has been applied to the problem domains of springs and systems of articulated parts. In this work we continue this line of work and extend it to the domain of soft materials in robotic soft hands.

GNN's have fueled a growing interest in learning models over graph structured data [15], these methods generally send messages over a fully connected graph, not explicitly capturing the connectivity in the underlying system. In this work we follow the model proposed in [11] which considers explicit system graph structure, that is, the connectivity of the graph mirrors the system connectivity.

C. Our Approach

We propose to learn a structured differentiable action-conditioned dynamics model based on an explicit joint connectivity graph, the "Sensorimotor Graph", of a robotic soft hand. By observing sequences of joint positions and actuation signals we infer the underlying system connectivity and then, to model the system dynamics, we condition our prediction model on the explicit connectivity graph structure.

We take advantage of the underlying structure of the system, modelling the system's Sensorimotor Graph as an explicit connectivity graph and then applying a Graph Neural Network (GNN)[16], [11] to model the system dynamics. This architecture has an implicit relational inductive bias by assuming a system composed of interacting parts [12], it learns over structured representations, parts connected by edges, invariant to order of nodes and compositional, prediction is result of many simple interactions.

Model predictive control formulations can then take the learnt differentiable system dynamics model as constraints and optimize for the actuation actions which minimize a control objective. [4].

Learning dynamics in this manner enables us to decouple control and model learning. This implies we can learn the system in a self-supervised setting, by "babbling" motor actions, that is, using a wide range of different control signals on the cable-driven actuator and observing the outcomes, avoiding costly reward samples. While we apply the methodology to the concrete example of a robotic soft hand, the method is general to any system of interacting parts, e.g.: springs [11], [17], cloth [10] or crowd behaviour [18].

D. Contributions

We formulate the problem of learning a sensorimotor model for a robotic soft hand's non-rigid kinematic chain as learning an explicit connectivity graph which we designate Sensorimotor Graph. Additionally, we propose the introduction of explicit actions to the graph forward model proposed in [11]. This enables the down-stream application of the learnt model as a differentiable system dynamics model in an optimal control application.

Furthermore, we apply this action conditioned graph forward model to the domain of a robotic soft hand, showing that it can capture the finger's coupled dynamics and comparing it to different non-structured baseline models. Lastly we benchmark the model to understand robustness to conditions with increasing adversity.

II. Problem Statement

In this work we look at modelling a robotic soft hand to obtain a differential dynamics model for an optimal control application. We assume the hand is a structured system of interacting parts. In a formal way, given system observations, $x_{1:t}$ and actuation signals $u_{1:t}$, predicting a sequence $x_{t:t+T}$ into the future. We decouple the problem in two parts, in the first we infer an explicit connectivity graph, or "Sensorimotor Graph", S, and a second part where we predict a future sequence, $x_{t:t+T}$, conditioned on S. We model the system dynamics as the interaction of N_c nodes represented as a Sensorimotor Graph, $S = (V, E)$, where V is the set of nodes, $V = \{v_k \in \mathcal{R}^n\}$, and E is the set of relations or edges between them $E = \{(x, y) \mid (x, y) \in V^2 \wedge x \neq y\}$.

The system state vector, x^t is given by the concatenation of the node positions, v_k^t. To be driven by external actuation signals we consider the actions as part of the node states, this allows us to either map specific actions to specific nodes, or map all actions to all nodes and the model inferring the relation between nodes and actuation signals.

The nodes interact through the system dynamics, F, such that, $v^{t+1} = F(x^t, E)$, we assume the case where the system dynamics can be factored in such a way that individual node state updates can be written as, $v_k^{t+1} = \sum_{i \in \{0,\dots,K \,:\, i \neq j\}} H(v_i^t, v_j^t, E_{(j,i)})$. That is, the nodes are related to other nodes through the edges between them. The edges represent the connectivity of the hand structure.

III. Model

As we have seen in the last section, in this work we consider a dynamical system with underlying Sensorimotor Graph structure. To best take advantage of this structure we will apply the Neural Relational Inference (NRI) model proposed in [11] which considers explicit system graph structure, that is, the connectivity of the graph mirrors the system connectivity.

The model is defined in two parts. A first part receives a sequence of system observations and infers the underlying edge structure and a second part which takes this edge structure and system state, predicting the system state into the future. Both these parts are trained jointly, either by providing ground truth labels for the edges or by learning these in an unsupervised manner end-to-end.

The first part, following the notation in [11], is called the encoder. The encoder takes a sequence of observed trajectories to infer pair-wise interactions. For this purpose, it takes a fully connected graph. On this graph, the interaction between nodes is captured by iterative node to edge and edge to node message passing iterations. The edge existence probability is captured as a distribution, $q_\phi(z|x)$, and the model is framed as a Variational Auto Encoder, for a more detailed discussion about the variational aspects of this model we refer the reader to [11].

More specifically, in the first pass the node to edge messages are computed using an embedding function, f_e, embedded node values from nodes i and j are then used to update the edge $e_{i,j}$ between them, after updating the edge state, another embedding function, f_v, takes the edge value and uses it to compute edge to node messages, the edge to node messages are aggregated for all edges connected to node i, then a final node to edge message passing operation infers the final edge value from which the edge existence probability is calculated using a softmax function.

$$\begin{aligned} & & \mathbf{h}_j^1 &= f_{\text{emb}}\left(\mathbf{x}_j\right) \\ v \to e: & & \mathbf{h}_{(i,j)}^1 &= f_e^1\left(\left[\mathbf{h}_i^1, \mathbf{h}_j^1\right]\right) \\ e \to v: & & \mathbf{h}_j^2 &= f_v^1\left(\sum_{i \neq j} \mathbf{h}_{(i,j)}^1\right) \\ v \to e: & & \mathbf{h}_{(i,j)}^2 &= f_e^2\left(\left[\mathbf{h}_i^2, \mathbf{h}_j^2\right]\right) \end{aligned} \tag{3}$$

Given the distribution, it is possible to either train the model in a supervised setting by providing ground truth labels for the edges, or training the model end-to-end by sampling edges from the inferred distribution and using these as graph structure for the subsequent trajectory prediction step. The NRI model uses a continuous approximation of the discrete distribution to effectively sample from a discrete distribution. Here a temperature parameter, τ, adjusts the "softness" of the resulting distribution, when $\tau \to 0$, the distribution converges to one-hot samples, as described in [11].

$$\mathbf{z}_{ij} = \operatorname{softmax}\left(\left(\mathbf{h}_{(i,j)}^2 + \mathbf{g}\right)/\tau\right) \tag{4}$$

The second part, called the decoder, takes the edges and uses them as explicit structure for the graph computations in the trajectory prediction step. Just like the encoder, the trajectory predictions are based on a sequence of node to edge and edge to node passes. An initial accumulation function updates the edge value by weighting adjacent nodes by the edge existence probability, $z_{i,j,k}$ represents the k-th entry in the of the vector $z_{i,j}$, and the embedding function, $\tilde{f}_e^k$ associated to this edge type k. Subsequently edges are aggregated in the nodes through an order invariant summation and transformed by an embedding function, $\tilde{f}_v$, added together with the current node state gives us the mean value, μ_j^{t+1} for the normal distribution from which the next node state, x_j^{t+1} is sampled.

$$
\begin{aligned}
v \rightarrow e: \quad & \tilde{\mathbf{h}}_{(i,j)}^t = \sum_k z_{ij,k} \tilde{f}_e^k \left(\left[\mathbf{x}_i^t, \mathbf{x}_j^t \right] \right) \\
e \rightarrow v: \quad & \boldsymbol{\mu}_j^{t+1} = \mathbf{x}_j^t + \tilde{f}_v \left(\sum_{i \neq j} \overline{\mathbf{h}}_{(i,j)}^t \right) \\
& p\left(\mathbf{x}_j^{t+1} \mid \mathbf{x}^t, \mathbf{z} \right) = \mathcal{N}\left(\boldsymbol{\mu}_j^{t+1}, \sigma^2 \mathbf{I} \right)
\end{aligned} \tag{5}
$$

The decoder can be structured as a transformation function modelled as an MLP or as an LSTM, acting recursively on the own internal state when updating the node using a memory cell.

IV. Experimental Setup

A. Data Collection

In order to collect diverse data from a soft hand gripper in motion, the Simulation Open Framework Architecture (SOFA) [3] is used, alongside with the Soft Robotics Toolkit [19]. The soft hand configuration is based on a cable actuated hand, where three-joint fingers are actuated by a pulling cable (Fig. 2). The finger base is fixed and the variation in the cable pull actuation signal allows the soft finger to experience a wide range of motion. Variability is added to the scene by including fingers in different number and configurations around a dodecagon. The actuation signal and elasticity of each finger vary in different trials. Each run in the SOFA environment creates a sequence of 100 sampled time steps. For each finger, three points are sampled, one for each joint.

B. Training and Validation Sets

A total of six different data sets are extracted from the soft gripper simulation. The training set for all tests includes only four fingers in a total of thirty different configurations (relative positions) and with four different values of elasticity we will reference this set as Trainset. For each sample, the type of motion applied to the cable pull actuation signal for every finger is the same, although the characteristics of the motion - amplitude, frequency and phase - are different and randomized. Three different types of motion are used, all resulting from operations between trigonometric functions. Our first set, Testset 0, includes novel configurations, some

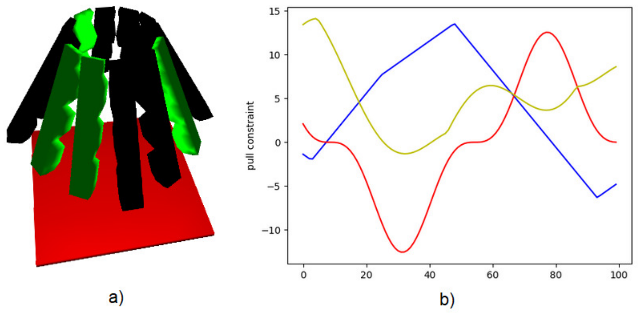

Fig. 2. a) Finger postions around the dodecagon with one configuration selected (green) b) Types of cable pull actuation signals used in the Trainset for 100 steps.

TABLE I
DESCRIPTION OF VALIDATION SETS

Test Sets	**Description**
Testset1	Unseen motions
Testset2	Unseen configurations
Testset3	Different number of fingers
Testset4	Shuffled points

combinations of motion and elasticity unseen at training time, as well as a different cable pull actuation signal.

We created four sets to test the model in different validation sets with varying conditions. The validation sets are setup to understand: Testset 1) How well the model generalizes to previously unseen motions, Testset 2) Effect of system configuration (relative position of joints) deviates from the training configurations, either due to sensor measurement errors or due to changes in the end-effector, Testset 3) Adaptation to changes in the number of fingers, either due to mechanical failure of one of the joints or sensor measurements failing to provide the position for any of the fingers or points, Testset 4) Robustness to changes in the order of the joint measurements, either due to tracking errors or wrong correspondences between detected parts.

More specifically the four validation sets are used to test the models generalization to motion, relative position, number of fingers and order of the points, respectively. The first test set, Testset 1, uses three different motions and includes random noise. The second test set, Testset 2, uses the same motions and elasticities as the training set but for thirteen unseen configurations. The third test set, Testset 3, uses only three-finger configurations with the same motions and elasticities whereas the fourth, Testset 4, and last validation set, shuffles the order of the twelve points for each sample.

C. Experiments

The model shall be assessed in two separate sets of experiments: comparison to other methods proposed in the literature and ablation studies to validate the robustness of the proposed approach under different conditions.

The first part looks at our approach in relation to baseline models. In this set of experiments the following models are quantitatively compared: 1) Linear, 2) Multi-layer Perceptron (MLP), 3) Long Short Term Memory sequence model (LSTM), 4) Neural Relational Inference (NRI).

We consider the linear and multi-layer perceptron models to validate if the task can be solved by simple linear or non-linear interpolation of the trajectory points. Long Short Term Memory model was chosen as being representative of a non-structured model, enabling us to compare the performance of a structured sequence model with a non-structured sequence model. The second part looks more closely at how different conditions affect the model under analysis. For this purpose, three models are considered - the two best performing models in the baselines and the structured model we selected to solve the task at hand.

V. Results and Discussion

A. Setup

The experiments are structured around two parts. The first, is a quantitative comparison of how the different models perform over the first validation set, Testset 0. The second, evaluates how each source of variability affects the different models and how they react to different types of generalization

B. Model Comparison

For the first experiment, the structure-based approach (NRI) is compared to three different baselines: Linear, MLP and LSTM. The NRI model has two model variations on the decoder part of the model, it can either be based on a MLP (NRI-mlp) or an RNN structure (NRI-rnn), in this experiment we consider both variants. For each of the possible decoder structures the encoder has two additional variations: supervised and unsupervised. The supervised encoder receives ground truth edges for the connections between the finger nodes, the unsupervised learns the connections in an end-to-end manner while learning to predict future trajectories.

TABLE II

Mean Squared Error for model and baselines

Model	**MSE 5**	**MSE 10**
Linear	7.987e-4	1.268e-3
MLP	7.878e-4	1.117e-3
LSTM	6.157e-4	1.354e-3
NRI-mlp	4.962e-4	1.336e-3
NRI-rnn	2.200e-4	**8.013e-4**
supervised NRI-mlp	4.753e-4	1.281e-3
supervised NRI-rnn	**2.108e-4**	7.457e-3

In this experiment all models are trained for 10 predictions steps. In the variations of the NRI with no ground truth edge information, a non-overlapping set of fifty time steps is provided to the encoder to estimate the connectivity graph. For all the baselines and for the NRI-rnn, there is an initialization or "burn-in" phase where the models have access to the five time steps before the sequence that is considered for the evaluation. The LSTM "burn-in" phase, similarly to the NRI, has fifty steps before the ten step prediction window. This allows the LSTM to initialize its internal state.

Table II shows how the proposed model compares to different baseline models, here we can see that the proposed model (NRI) outperforms the baseline models in the first testset, Testset 0, which has a diverse set of motions, configurations and elasticities, when the prediction step equals the training step. Within the different variations of the NRI model we find that the best combination is the unsupervised variant (NRI-rnn). We will look more closely at how this NRI model variant performs under different conditions in the next experiment.

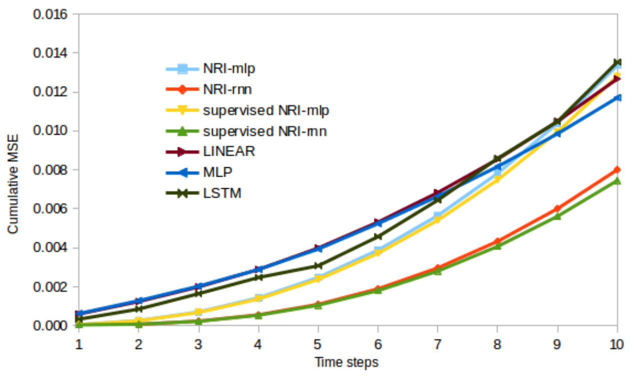

Fig. 3. Cumulative mean square error (MSE) over 10 time-steps for different models.

Figure 3 shows how the models compare when predicting for 10 steps ahead, the same number they were trained on. This validation set includes small variations in motion, elasticity and configurations. We can see that the NRI model using an RNN decoder outperforms the other variations considerably. We notice that while the NRI-rnn has better performance for the first steps, particularly relevant in planning, the accumulated error starts deteriorating as we predict beyond the training distribution, further into the future. We will look at this more closely in the ablation studies.

C. Ablation Study

We will now look at the second set of experiments (Fig. 4) which seek to see how the model performs under different input distribution shifts. To test the models under a diverse set of conditions, we will consider Testset 0 under a prediction step of 25 to test how well the model reacts to auto-regressively predicting more steps ahead than it was trained on and the four test-sets introduced in the experimental setup, refer to Table 1 for an overview of the different test-sets. This enables us to study the effects of the different sources of prediction error separately. For this ablation study we will consider the best performing NRI model and two relevant baselines, LSTM and MLP.

From the four ablation studies, we can see that the NRI performs worse than the baseline models when generalizing to the distribution shift introduced by predicting longer sequences. In this first Testset, the MLP seems to be capable of predicting longer sequences with a lower mean error, the

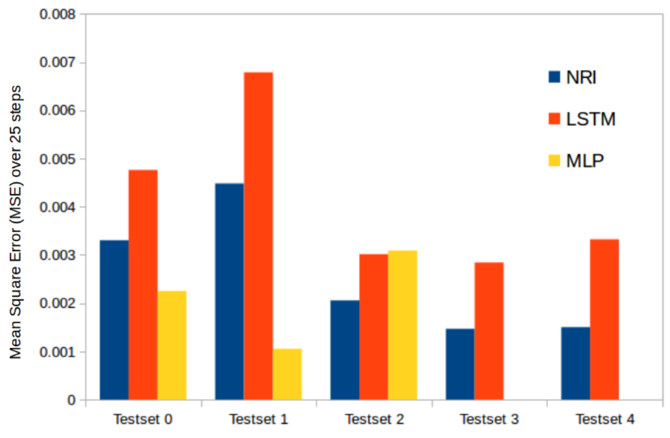

Fig. 4. Mean square error (MSE) after the 25 timesteps for different models and validation sets.

model also proves to have good generalization to new motions. This advantage of the MLP compared to the structure-based models can be justified with the fact that it is able to exploit the relations between the nodes in adjacent positions of the input vector. The NRI, due to it's order invariant graph-based structure only, has access to the nodes without explicit order like in a vector based representation of the nodes.

This, however, makes the MLP quite fragile to changes or disturbances in the data. Testing the same model in a situation with new relative positions of the fingers, MLP's prediction error increases considerably. Under the same conditions, we can see the NRI outperforming the baseline models. Under the condition of changing the number of fingers, the MLP is not able to generalize and has an error significantly higher than the other models and is therefore not considered in this validation set. We can again see the NRI outperforming the baseline models in this condition. The last test-set considers the situation where points are shuffled and the points at test are in a different order than the configurations it was trained on. This models a tracking or re-identification error. Here, again, we can see that, due to its order invariance properties, the NRI model outperforms the baseline models.

D. *Discussion*

We evaluated the model against different baseline models in two different sets of experiments, in the first we found that the proposed action conditioned NRI model outperforms the baseline models on Testset 0 which tests for small variations in system attributes, this validates the robustness of the proposed model, furthermore from this first set of experiments we can see that the unsupervised variant of the NRI model outperforms the other variants, hinting that providing ground truth edge connectivity in complex non-rigid kinematic chains performs worse than self-supervised discovery of edges, which might find non-trivial edge connectivities not captured in the ground truth edge labels.

In the second set of experiments we found that the NRI model performs worse than the baseline models when tested to auto-regressively predicting longer sequences than trained on and under variations in motion. When tested under variations of configuration and number of joints it outperforms the baseline models, validating the robustness to possible sensor noise. Additionally when tested on a test-set of shuffled points it shows it is able to reconstruct the connectivity and outperform the baseline models, validating robustness to re-identification or tracking issues.

VI. Conclusions and Future Work

In this work we propose applying a graph neural network- the "Sensorimotor Graph" - to the domain of modelling a robotic soft hand. More specifically we described the underlying compositional assumptions of the system and showed how a structured graph neural network successfully satisfies the model assumptions. We denote our model as the "Sensorimotor Graph" as it is inspired on the way human infants acquire sophisticated motor capabilities, e.g. of the human hand, through sensorimotor learning and motor babbling.

We chose the Neural Relation Inference as a candidate graph neural network for the problem domain. We showed how the models explicit graph representation enables us to infer the systems underlying structure. To evaluate the model, we benchmarked it against non-structured baselines and evaluated the model robustness to gradually more adverse conditions. We found that the model performs well under conditions which require generalizing to changes in structure.

More specifically it performed well under conditions where the number and location of the fingers changed.

Furthermore, we looked at different variations of the model and observed that out of the two encoder variations, supervised and unsupervised, the unsupervised edge discovery sometimes performs better. This means that explicitly providing the system constraints does not necessarily lead to optimal results. Training the edge structure inference and dynamics prediction jointly creates a non-natural hand structure with finger to finger connections similar to the hand to hand interactions in other motion capture modeling approaches [20], [11].

We showed that graph neural networks, more specifically the Neural Relational Inference model, can be used to robustly model a system of interacting soft material parts in the form of a differentiable dynamics model. This opens up promising avenues for combining the learnt model with a non-convex optimal control framework, using the differentiable dynamics model as systems constraints.

We believe that the encouraging results and generalization capabilities of our "Sensorimotor Graph" model, pave the way to the ability to acquire the structure and dynamics of complex systems, such as the human/robotic hand, and will be of utmost importance for the deployment of complex, non-rigid kinematic chains and articulated end-effectors such as soft hands for advanced manipulation.

Acknowledgment

This work was supported by a doctoral grant from FCT (SFRH/BD/139092/2018).

References

[1] C. von Hofsten, "An action perspective on motor development," *Trends in cognitive sciences*, vol. 8, no. 6, p. 266—272, June 2004.

[2] B. S. Homberg, R. K. Katzschmann, M. R. Dogar, and D. Rus, "Haptic identification of objects using a modular soft robotic gripper," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 1698–1705.

[3] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, and S. Cotin, *SOFA: A Multi-Model Framework for Interactive Physical Simulation*, 06 2012, vol. 11.

[4] B. Amos, I. Jimenez, J. Sacks, B. Boots, and J. Z. Kolter, "Differentiable mpc for end-to-end planning and control," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 8289–8300. [Online]. Available: http://papers.nips.cc/paper/8050-differentiable-mpc-for-end-to-end-planning-and-control.pdf

[5] J. L. Pons, R. Ceres, and F. Pfeiffer, "Multifingered dextrous robotics hand design and control: A review," *Robotica*, vol. 17, no. 6, p. 661–674, Nov. 1999.

[6] T. Hsia, "Adaptive control of robot manipulators - a review," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3, 1986, pp. 183–189.

[7] F. Ebert, C. Finn, A. X. Lee, and S. Levine, "Self-supervised visual planning with temporal skip connections," in *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, ser. Proceedings of Machine Learning Research, vol. 78. PMLR, 2017, pp. 344–356.

[8] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *ICML*, 2019.

[9] J. Kober, J. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, pp. 1238 – 1274, 2013.

[10] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," in *ICLR*, 2019.

[11] T. Fetaya, E. Wang, K.-C. Welling, M. Zemel, T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, "Neural relational inference for interacting systems," *arXiv: Machine Learning*, 2018.

[12] P. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, Çaglar Gülçehre, H. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. R. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," *ArXiv*, vol. abs/1806.01261, 2018.

[13] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia, "Learning to simulate complex physics with graph networks," *ArXiv*, vol. abs/2002.09405, 2020.

[14] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ser. ICML'17, 2017, p. 1263–1272.

[15] S. van Steenkiste, M. Chang, K. Greff, and J. Schmidhuber, "Relational neural expectation maximization: Unsupervised discovery of objects and their interactions," in *6th International Conference on Learning Representations*, 2018.

[16] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[17] N. Watters, A. Tacchetti, T. Weber, R. Pascanu, P. Battaglia, and D. Zoran, "Visual interaction networks: Learning a physics simulator from video," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 4542–4550.

[18] S. H.Y., G. Shivakumar, and H. S. Mohana, "Crowd behavior analysis: A survey," in *2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT)*, 2017, pp. 169–178.

[19] D. Holland, E. Park, P. Polygerinos, G. Bennett, and C. Walsh, "The soft robotics toolkit: Shared resources for research and design," *Soft Robotics*, vol. 1, pp. 224–230, 09 2014.

[20] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.