Learning to Predict the Outcome of Planar Manipulation of an Unknown Shape

Harry Clark
School of Computer Science
University of Birmingham

Jeremy L Wyatt
School of Computer Science
University of Birmingham

***Abstract*—Understanding how an object will move under manipulation comes naturally to humans. Encoding these mechanical laws analytically is challenging. Recently, data-driven approaches have been applied to learn predictive models of motion. In this paper we present a method to generalise predictions of motion across different object shapes. This uses a novel descriptor of the combined shape and pose as input to a neural network predictor. The predictor is a Mixture Density Network (MDN) and so it gives a measure of uncertainty in the action outcome. The system is evaluated by testing its ability to generalise its predictions to unknown 2D shapes after training with simpler shapes.**

I. Introduction

Central to object manipulation in humans is the ability to predict the outcomes of manipulation actions. In robotics, attempts have been made to apply the analytic laws of mechanics to perform such predictions. This is hard because such models make approximations and require precise knowledge of parameters that are challenging to estimate online, such as frictional coefficients, coefficients of restitution, and mass distributions. An alternative is to learn a predictive (forward) model of object motion from experience. This paper shows how to use a neural network learner with Gaussian output units to learn a probabilistic predictive model based on sensorimotor input-output pairs. This model can predict the uncertainty in action outcomes and generalise to unknown objects.

We know that the human motor system can learn to model the complex non-linear relationships between motor action, object properties and pose, so as to predict the motion of a manipulated object. There is evidence that the cerebellum contains just such a forward model; both predicting the consequences of motor actions and detecting errors in these predictions [1], [2], [3], [4]. Repeating cerebellar units comprise a modularised and highly parallel architecture [5]. It has been suggested that this is an efficient strategy for modelling sensorimotor prediction by making modules context-specific [6], [7], [8], that is, modules learn simple features which collectively predict complex motions. Generalisation to novel conditions such as unknown object shape thus uses a combination of relevant features. Inspired by this, here we explore the application of a mixture density network (MDN) [9] to the problem of prediction learning. This variant has the added benefit of modelling the uncertainty in an action outcome.

II. Related Work

Modelling object behaviour under action can be tackled by the application of the laws of mechanics, either using quasi-static assumptions or incorporating dynamics [10], [11], [12], [13], [14]. This is challenging, due to the approximations made by simulators and the difficulty of estimating necessary parameters. Another approach to the prediction problem is based on data-driven learning. This has previously been applied to planar pushing [13], [15], [16], [17], [18] as well as rigid-body motions [19]. Data based methods are gaining traction as larger data-sets become available [20]. We employ the high-fidelity planar pushing data-set provided by Yu et al. [21].

Neural networks have already been used to model 3D rigid body transformations in several ways, including the 'physical intuition of object manipulation [22], [23], [24], ball motion [25], stability of stacked blocks [26] and object dynamics in static images [27]. Of the object manipulation work conducted, there has been some limited success in generalisation across scenes [23] and objects [24]. All research has used convolutional filters with high-dimensional input data, typically in the form of images. The data-set we employ naturally allows a lower dimensional representation of shape.

Non-neural machine learning approaches [19] include using a product of kernel density estimates (KDE) to predict 3D rigid body transformations under push actions. This approach afforded a degree of shape generalisation. Another approach is the use of Heteroscedastic Gaussian Process models for prediction [28], or for prediction and push planning [29]. In these two cases the methods are trained to predict on individual objects and show no capacity for shape generalisation

III. Deep Learning with Mixture Density Networks

A. Formal Definitions

Multi-layer perceptrons (MLPs) are global function approximators suitable for modelling non-linear vector functions. MDNs are a variation that support any neural architecture with a feed-forward structure. The stochasticity of planar push effects means that to correctly model object manipulation, the learned network would ideally generate an output density; thereby modelling the uncertainty of a given manipulation outcome. Using a mixture model [30], any general distribution can be modelled in terms of probability densities of target data being represented as linear combinations of kernel functions in the form

$$p(t|x) = \sum_{i=1}^{m} \alpha_i(x)\phi_i(t|x) \tag{1}$$

where t represents a target vector which is conditioned on an input vector x, m is the number of components in the mixture.

α denotes mixing coefficients which act as prior probabilities conditioned on x of the target vector t being generated by the i^{th} mixture component. ϕ represents the conditional density of the target vector for the i^{th} kernel. The Gaussian kernel is given by the form

$$\phi_i(t|x) = \frac{1}{(2\pi)^{\frac{c}{2}}\sigma_i(x)^c} \exp\left\{\frac{\|t-\mu_i(x)\|^2}{2\sigma_i(x)^2}\right\} \quad (2)$$

where vector μ_i represents the i^{th} kernel centre, with variance σ_i and c denotes the dimensionality of the target vector.

As the mean, variance and mixing coefficients are general continuous functions of x, these can be modelled as outputs of an MLP that uses vector x as an input [9]. Implementing this merely requires replacement of the least-squares error by the negative log likelihood

$$E^q = -ln\left\{\sum_{i=1}^{m} \alpha_i(x^q)\phi_i(t^q|x^q)\right\} \quad (3)$$

for pattern q. This is the same error function used in Jacobs et al's [31] model of competing local experts.

B. Training and Predicting Planar Manipulation

In this paper, we use the high fidelity experimental planar pushing data-set of Yu et al. [21]. Simple, flattened 3D shapes (limited to transformations on the plane) were pushed by a cylindrical robotic finger. For a full description of the shapes and setup, see [21] and [32].

We represent motion of the pusher and the object as 2D rigid body transformations. A 2D rigid body transform $[R, t]$ can be specified by a 3-parameter axis-angle transform made up of a single rotation $R \in$ SO(2) and a translation $t \in \mathbb{R}^2$. Using TensorFlow we trained an MDN using the adaptive moment estimation (ADAM) optimiser to predict the transformation of the object from its initial pose, pusher trajectory and the average forces exerted on the pusher. Several techniques employed to improve generalisation included L1 weight regularisation, early stopping and dropout [33]. A schematic of the network architecture is provided in Figure 1. Outputs were generated with a convex combination of kernel means weighted by network output mixing coefficients and averaged across 500 dropout permutations.

IV. Modular Prediction with Action Generalisation

In this experiment we studied the ability of learned models to predict the effect of a push for a specific object. Each learner was split into modules, one for each object, following the principles outlined in [19], [8] for modularisation. Thus, generalisation was over the action space: push location and push direction. We created this architecture seperately with an MDN, a MLP with sigmoid units trained on squared error loss and a kernel density estimator (KDE) for comparison. We implemented the KDE following [34], choosing a bandwidth according to Silverman's rule of thumb. The appeal of KDE is that it offers greater data efficiency and avoids extensive parameter tuning. Figure 2 shows the performance comparison. The MDN and MLP outperforms the KDE and thus advocates the use of neural networks for action generalisation, however the mixture model implemented in the MDN did not improve action generalisation, suggesting its redundancy in this case.

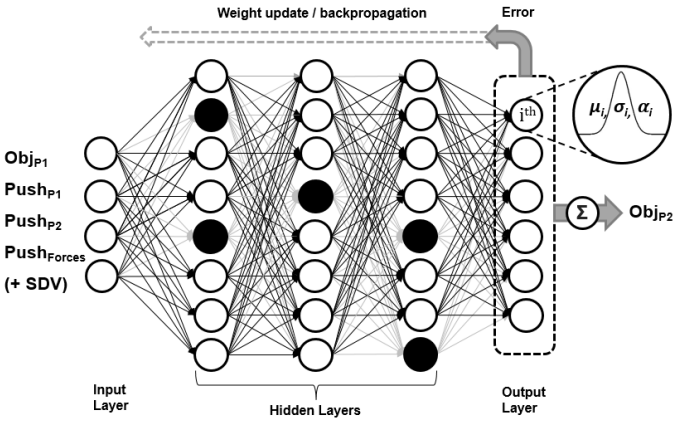

Fig. 1: Neural Architecture for Transformation Prediction. Inputs included initial object and pusher pose (P1), end pusher pose (P2), average force/torque and (for shape generalisation) the SDV. Module outputs (e.g. ith) are combined using a convex combination and averaged over permutations of the dropped out network generating a predictive density of object end pose. One such dropout permutation is shown.

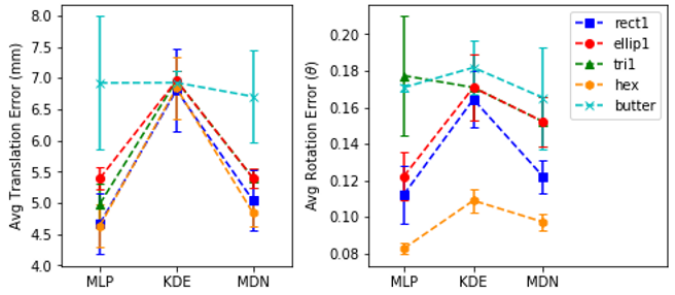

Fig. 2: Displacement predictions of regression neural networks, kernel density estimation and mixture density networks trained on each shape individually. Error bars denote SEM.

V. Prediction with Shape Generalisation

The first experiment shows an ability to generalise with respect to action, but not with respect to object shape. To achieve shape generalisation, we could use a variety of representations. We draw on ideas from the neuroscience literature that suggest that internal models are typically influenced by an egocentric frame of reference. In addition, while we could use the full image as an input vector it is sensible to assume some simple edge based processing to reduce the dimensionality and increase the invariance of the input space. To this end we introduce a shape descriptive vector (SDV), as well as using mass and moment of inertia as part of the input space.

The SDV has a polar frame of reference that is influenced by both the object's position and the agent's viewing position. The origin of the polar axis is the visual centre of mass of the object. The direction of the polar axis is upwards in the image frame. The SDV is a vector of polar coordinates in this frame, obtained by sampling the boundary of the shape at regular angular intervals, in its current orientation as projected onto the image plane. The start point and end point occur in the direction of the polar frame. We used a sampling rate (Figure 3)

of 15°, creating a 24 dimensional vector. The SDV captures both the orientation relative to the observer and the shape of the object in a single representation. To achieve prediction with shape generalisation the SDV may serve as part of the input space for any of the learners specified above.

To test the hypothesis that the SDV would enable good generalisation of predictions with respect to shape and orientation we retrained an MDN, but now using one network for all training shapes and the SDV as part of the input vector (SDV-MDN). Training was performed using rectangles (rect2-3), ellipses (ellip2-3), triangles (tri2-3) and the hexagon. We then tested predictions on an unseen hold-out set of shapes (rect1, ellip1, tri1, butter). The results are shown in Figure 3. We present the prediction error in translation (mm) and rotation (rads) plotted against the number of output kernels in the MDN. The bottom panels show the prediction of motion for a specific action for each test shape with the ground truth marked.

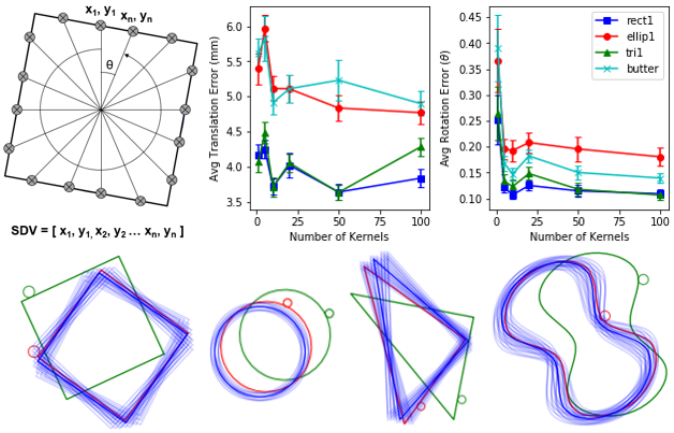

Fig. 3: SDV Design and Shape-Generalised Predictions using SDV-MDN. Plots show motion predictions with increasing numbers of kernels in the output layer of the MDN. The SDV is a vector of polar coordinates sampled at shape borders conveying shape and orientation. Error bars denote SEM. The mean predictions (shapes with bold blue lines) came from a network trained on samples from rect2-3, ellip2-3, tri2-3 and hex. Uncertainty (thin blue lines), shifted by one standard deviation in each direction from the mean for x, y and θ. The initial (green) and ground truth final (red) object pose and pusher position are shown.

A. Prediction of Uncertainty in Object Motion

In the data-set the same action, when repeated, results in a distribution of displacements [21]. We examined the ground truth versus predicted uncertainty in push outcome using 100 repeated pushes on one side of the 'rect1' shape (at every contact position and pushing angle). Comparing the predicted displacement from the MDN with the observed (ground truth) displacement shows that the network can capture this uncertainty for planar pushes. Figure 4 shows the mean and standard deviation of displacement of the x, y and θ predicted by the MDN compared with the ground truth. This shows an ability of the MDN not only to generalise the prediction of the mean motion with respect to shape, but also to generalise the predicted uncertainty with respect to shape.

VI. Discussion

In this paper we introduced MDNs as a method to model push mechanics inspired by cerebellar neuro-circuitry. Our results are comparable to existing methods [28] that use variational heteroscedastic Gaussian processes (VHGP) on the same data. Normalised mean square error (NMSE) of our shape generalisation model ranged from 0.4-0.6, marginally worse than the 0.37 reported error using VHGPs [28]. This reflects the greater complexity of making predictions under shape-generalisation. Further, increasing module outputs within the MDN improved prediction. Such improvements were not seen with action generalisation in Figure 2 (MLP vs MDN). This likely highlights the benefit of modularisation in cases of greater complexity. Our model also outperformed KDE.

We introduced a simple combined shape and orientation representation. This affords accurate shape generalisation, even with complex shapes such as 'butter'. This representation is akin to intrinsic frames of reference, that may be utilised by humans for predicting manipulation dynamics [35]. Learning with MDNs and shape-pose generalisation offers a way to build forward models for object manipulation. As shown here, forward models can be trained to predict the displacement of objects from planar pushes and express uncertainty in this motion.

References

[1] O. Hikosaka, K. Nakamura, K. Sakai, and H. Nakahara, "Central mechanisms of motor skill learning," *Current Opinion in Neurobiology*, vol. 12, no. 2, pp. 217–222, 2002.

[2] J. W. Krakauer and P. Mazzoni, "Human sensorimotor learning: adaptation, skill, and beyond," *Current Opinion in Neurobiology*, vol. 21, no. 4, pp. 636 – 644, 2011.

[3] V. Penhune and C. Steele, "Parallel contributions of cerebellar, striatal and m1 mechanisms to motor sequence learning," *Behav. Brain Res.*, vol. 226, pp. 579–91, 2012.

[4] R. Shadmehr and J. W. Krakauer, "A computational neuroanatomy for motor control," *Exp. Brain Res.*, vol. 185, pp. 359–381, 2008.

[5] T. Ruigrok, "Ins and outs of cerebellar modules," *The Cerebellum*, vol. 10, no. 3, pp. 464–474, Sep 2011.

[6] N. Ramnani, "The primate cortico-cerebellar system: Anatomy and function," vol. 7, pp. 511–22, 08 2006.

[7] M. Ito, "Control of mental activities by internal models in the cerebellum," vol. 9, pp. 304–13, 05 2008.

[8] M. Haruno, D. M. Wolpert, and M. Kawato, "Mosaic model for sensorimotor learning and control," *Neural computation*, vol. 13, no. 10, pp. 2201–2220, 2001.

[9] C. M. Bishop, "Mixture density networks," Tech. Rep., 1994.

[10] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, February 1987.

[11] M. Dogar and S. Srinivasa, "A planning framework for non-prehensile manipulation under clutter and uncertainty," *Autonomous Robots*, vol. 33, no. 3, pp. 217–236, Oct 2012.

[12] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *Int. J. Rob. Res.*, vol. 5, no. 3, pp. 53–71, Sep. 1986.

[13] M. Salganicoff, G. Metta, A. Oddera, G. Sandini, M. Salganico, G. Metta, A. Oddera, and G. Sandini, "A vision-based learning method for pushing manipulation," in *In AAAI Fall Symposium Series: Machine Learning in Vision: What Why and*, 1993.

[14] A. Cosgun, T. Hermans, V. Emeli, and M. Stilman, "Push planning for object placement on cluttered table surfaces," pp. 4627–4632, 09 2011.

[15] S. Walker and J. K. Salisbury, "Pushing using learned manipulation maps," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 3808–3813.

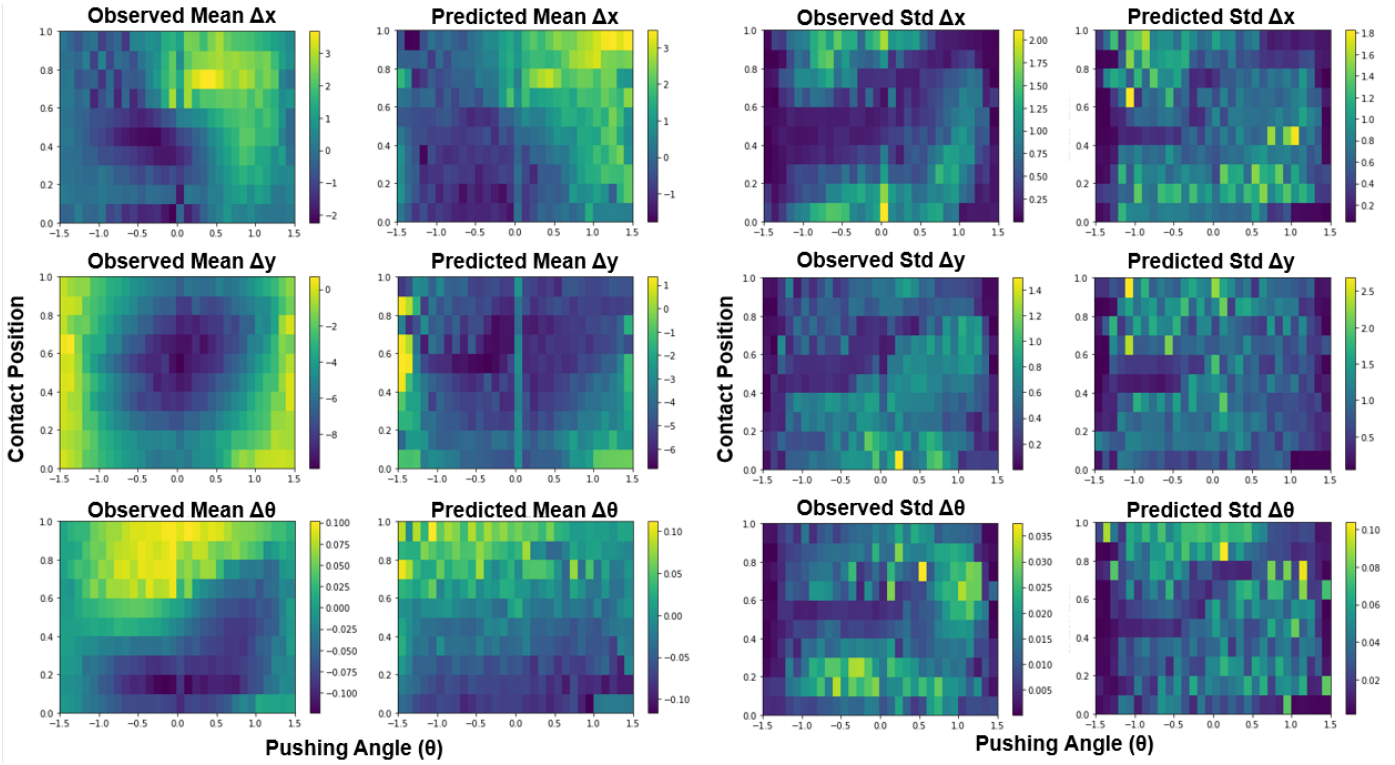

Fig. 4: Shape generalisation for mean and standard deviation of motion. The first two columns show the mean motion $(\Delta x, \Delta y, \Delta\theta)$ predicted by the SDV-MDN (column 1) for a range of contact positions and pushing angles compared to the ground truth (column 2). The third and fourth columns show the standard deviation in the motion predicted by the SDV-MDN (column 4) compared to the ground truth (column 3). Units are in mm and radians.

[16] M. Lau, J. Mitani, and T. Igarashi, "Automatic learning of pushing strategy for delivery of irregular-shaped objects," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 3733–3738.

[17] T. Meriçli, M. Veloso, and H. L. Akın, "Push-manipulation of complex passive mobile objects using experimentally acquired motion models," *Autonomous Robots*, vol. 38, no. 3, pp. 317–329, Mar 2015.

[18] J. Zhou, R. Paolini, J. A. Bagnell, and M. T. Mason, "A convex polynomial force-motion model for planar sliding: Identification and application," *CoRR*, vol. abs/1602.06056, 2016.

[19] M. Kopicki, S. Zurek, R. Stolkin, T. Moerwald, and J. L. Wyatt, "Learning modular and transferable forward models of the motions of push manipulated objects," *Autonomous Robots*, vol. 41, no. 5, pp. 1061–1082, 2017.

[20] Y. Huang, M. Bianchi, M. Liarokapis, and Y. Sun, "Recent data sets on object manipulation: A survey," *Big Data*, vol. 4, pp. 197–216, 2016.

[21] K. Yu, M. Bauzá, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed: A high-fidelity experimental data set of planar pushing," *CoRR*, vol. abs/1604.04038, 2016. [Online]. Available: http://arxiv.org/abs/1604.04038

[22] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," *CoRR*, vol. abs/1505.01596, 2015. [Online]. Available: http://arxiv.org/abs/1505.01596

[23] A. Byravan and D. Fox, "Se3-nets: Learning rigid body motion using deep neural networks," *CoRR*, vol. abs/1606.02378, 2016. [Online]. Available: http://arxiv.org/abs/1606.02378

[24] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," *CoRR*, vol. abs/1610.00696, 2016. [Online]. Available: http://arxiv.org/abs/1610.00696

[25] K. Fragkiadaki, P. Agrawal, S. Levine, and J. Malik, "Learning visual predictive models of physics for playing billiards," *CoRR*, vol. abs/1511.07404, 2015. [Online]. Available: http://arxiv.org/abs/1511.07404

[26] A. Lerer, S. Gross, and R. Fergus, "Learning physical intuition of block towers by example," in *Proc. of the 33rd ICML*, 2016, pp. 430–438.

[27] R. Mottaghi, H. Bagherinezhad, M. Rastegari, and A. Farhadi, "Newtonian image understanding: Unfolding the dynamics of objects in static images," *CoRR*, vol. abs/1511.04048, 2015. [Online]. Available: http://arxiv.org/abs/1511.04048

[28] M. Bauzá and A. Rodriguez, "A probabilistic data-driven model for planar pushing," *CoRR*, vol. abs/1704.03033, 2017. [Online]. Available: http://arxiv.org/abs/1704.03033

[29] E. Arruda, M. J. Mathew, M. Kopicki, M. Mistry, M. Azad, and J. L. Wyatt, "Uncertainty averse pushing with model predictive path integral control," in *IEEE Humanoids*. IEEE, 2017, pp. 497–502.

[30] G. Mclachlan and K. Basford, *Mixture Models: Inference and Applications to Clustering*, 01 1988, vol. 38.

[31] R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton, "Adaptive mixtures of local experts," *Neural Comput.*, vol. 3, no. 1, pp. 79–87, Mar. 1991. [Online]. Available: http://dx.doi.org/10.1162/neco.1991.3.1.79

[32] M. Lab. Website for push dataset. [Online]. Available: http://mcube.mit.edu/ push-dataset

[33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[34] M. Kopicki, S. Zurek, R. Stolkin, T. Mrwald, and J. Wyatt, "Learning to predict how rigid objects behave under simple manipulation," in *2011 IEEE ICRA*, May 2011, pp. 5722–5729.

[35] A. A. Ahmed, D. M. Wolpert, and J. R. Flanagan, "Flexible representations of dynamics are used in object manipulation," in *Current Biology*, 2008.