Robots that Anticipate Pain: Anticipating Physical Perturbations from Visual Cues through Deep Predictive Models

Indranil Sur[1†] and Heni Ben Amor[2]

***Abstract*— To ensure system integrity, robots need to proactively avoid any unwanted physical perturbation that may cause damage to the underlying hardware. In this paper, we investigate a machine learning approach that allows robots to anticipate impending physical perturbations from perceptual cues. In contrast to other approaches that require knowledge about sources of perturbation to be encoded before deployment, our method is based on experiential learning. Robots learn to associate visual cues with subsequent physical perturbations and contacts. In turn, these extracted visual cues are then used to predict potential future perturbations acting on the robot. To this end, we introduce a novel deep network architecture which combines multiple sub-networks for dealing with robot dynamics and perceptual input from the environment. We present a self-supervised approach for training the system that does not require any labeling of training data. Extensive experiments in a human-robot interaction task show that a robot can learn to predict physical contact by a human interaction partner without any prior information or labeling.**

I. INTRODUCTION

According to Isaac Asimov's third law of robotics [1], "a robot must protect its own existence as long as such protection does not conflict with the First or Second Law", i.e., as long as it does not harm a human. Aspects of safety and self-preservation are tightly coupled to autonomy and longevity of robotic systems. For robots to explore their environment and engage in physical contact with objects and humans, they need to ensure that any such interaction may not lead to tear, damage, or irreparable harm to the underlying hardware. Situations that jeopardize the integrity of the system need to be detected and actively avoided. This determination can be performed in either a reactive way, e.g., by using sensors to identify forces acting on the robot, or in a pro-active way, e.g., by detecting impending collisions. In recent years, a plethora of safety methods have been proposed that are based on reactive strategy. Approaches for compliant control and, in particular, impedance control techniques have been shown to enable safe human-robot interaction [2] in a variety of close-contact tasks. Such methods are typically referred to as *post-contact* approaches, since they react to forces exchanged between the system and its environment after they first occur.

In many application domains, however, robots need to pro-actively reason about impending damage before it occurs.

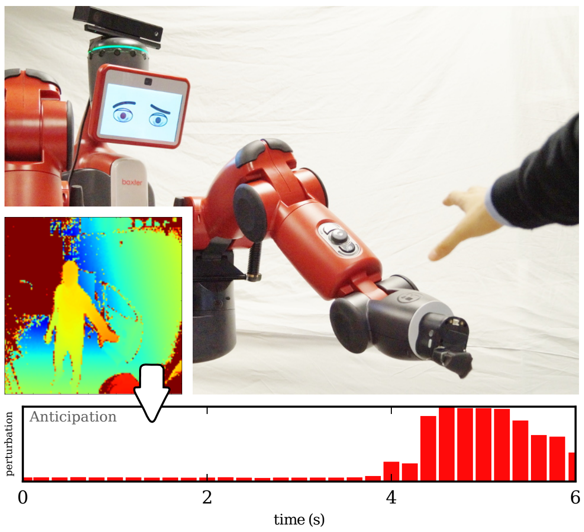


Fig. 1: Baxter robot anticipates physical contact by human.

Methods that tackle these scenarios, have mostly focused on proximity detection and collision avoidance. Motion tracking or other sensing devices are used to identify nearby humans and obstacles in order determine whether they intersect the robot's path. To this end, a human expert has to reason about the expected obstacles before deployment and, in turn, hand-code methods for object recognition, human tracking, or collision detection. Such methods are largely based on the status quo of the environment and do not take in account the expected future states. For example, the behavior of a human interaction partner may already provide cues whether or not physical contact is to be expected. In addition, such methods suffer from limited adaptivity – the robotic system is not able to incorporate other or new sources of physical perturbations that have not been considered by the human expert. Changes in the application domain typically require the expert to specify the set of obstacles/perturbants and how they can be identified from sensor data. However, for robots to autonomously explore new goals, tasks, and environments they cannot be constantly relying on human intervention and involvement. In order to increase resilience of robotic systems, the processes responsible for ensuring safety should inherently be (a) adaptive to changes that occur during the cycle of operation and (b) anticipative in nature, so as to proactively avoid damage.

In biology such processes are common place: humans and animals experience *pain* as the guiding signal which ensures self-preservation and safe, "open ended" exploration of the

[1]Indranil Sur is with the Center for Vision Technologies, SRI International, Princeton, NJ 08540 indranil.sur@sri.com

[2]Heni Ben Amor is with the School of Computing, Informatics, Decision Systems Engineering, Arizona State University, 699 S Mill Ave, Tempe, AZ 85281 hbenamor@asu.edu

[†] Author contributed to this work as his Master's Thesis at Arizona State University.

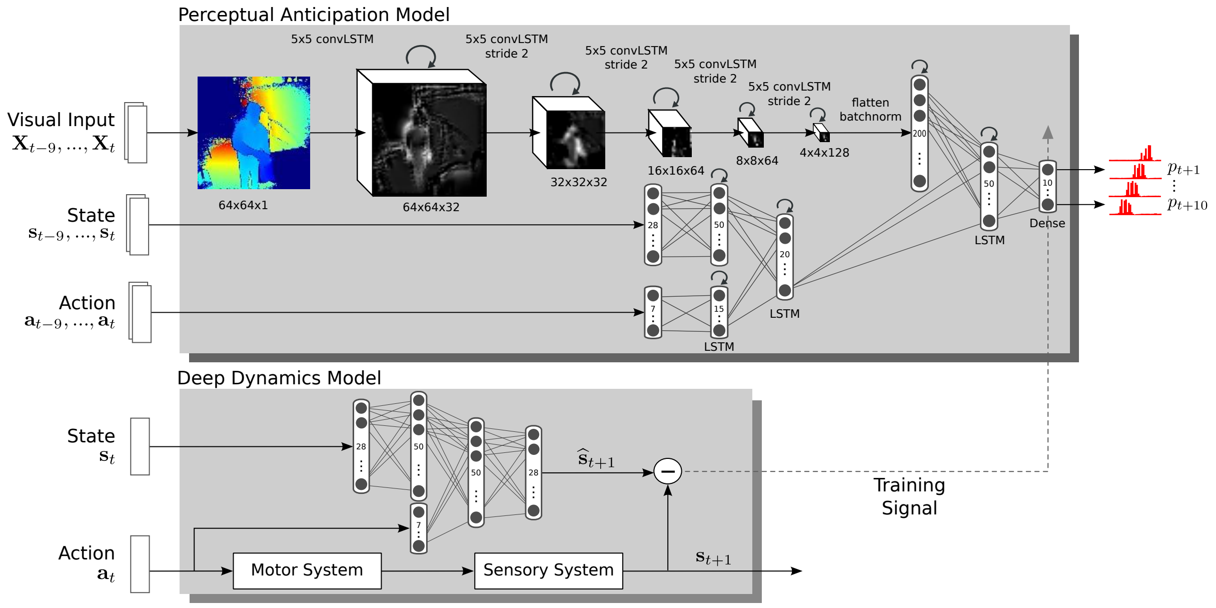


Fig. 2: Architecture of the Deep Predictive Model for perturbations.

world. Over time, biological creatures *learn to anticipate* impending pain sensations from environmental and proprioceptive cues, e.g., from the velocity and direction of an obstacle, or an imbalance in posture. Repeated exposure to a sensory cue (e.g. a specific image or object) preceding a negative outcome (e.g. electric shock) will turn the cue into a conditioned stimulus that helps anticipate the shock in future trials. This ability to associate a certain stimuli with negative future sensations is considered to be essential to survival.

Inspired by the relationship between pain and learning in biological creatures, we propose in this paper a new method for learning to anticipate physical perturbations in robotic systems. Robot safety is realized through an adaptive process in which prior experiences are used to predict impending harm to the system. These predictions are based on (1) environmental cues, e.g., camera input, (2) proprioceptive information, and (3) the intended actions of the robot. We introduce a *deep predictive model* (DPM) that leverages all three sources of information in order to anticipate contact and physical perturbations in a window of future time steps.

We will evaluate the introduced system in a set of experiments in which a robot has to anticipate physical perturbations caused by a human interaction partner. We will show that the introduced model effectively anticipates contact using either RGB or RGB-D camera sensors. While the introduced method can be used to actively avoid noxious states, we will focus our analysis in this paper to the detection of perturbations only.

II. Related Work

Safety plays a critical role in the field of robotics. Recently, various methods have been put forward in order to protect a human interaction partner from harm. The work in [3], for example, uses proprioceptive sensing to identify collisions and, in turn, execute a reactive control strategy that enables the robot to retract from the location of impact. In a similar vein, the work in [4] describes methods for rapid collision detection and response through trajectory scaling. Many approaches to safe human-robot interaction are based on rapid sensing through force-torque sensors or tactile sensors [5]. Another approach is to generate estimates of external forces acting on a robot using disturbance observers [6]. These approaches, however, require a model of the underlying plant, which in the case of complex humanoid robots can become challenging to derive. In addition, nonlinearities underlying the current robot or task can often lead to instabilities in the system [7]. To circumvent such challenges, several approaches have been proposed for learning perturbation filters using a data-driven machine learning method [8], [9], [10]. An alternative, bio-inspired approach was proposed in [11]. In particular, an “Artificial Robot Nervous System” was introduced which emulates the human skin structure, as well as the spikes generated whenever an impact occurs.

All of the above techniques are reactive in nature. Only after contact with its environment, can a robot detect a physical impact or perturbation and react to it. However, many critical situations require a proactive avoidance strategy. To this end, various methods for human motion anticipation have been proposed [12], [13], [14]. Given a partially observed human motion, a robot can anticipate the goal location and intermediate path and accordingly generate a collision free navigation strategy. However, such approaches are brittle in that they require some form of human tracking. If the source of the perturbation is non-human, e.g., a moving object then no anticipation can occur. In this paper, we are interested in dynamic approaches to anticipation of perturbations. Robots learn to predict contacts or impact by associating them with visual cues. This is similar in spirit to [15] where dashcam videos were used to predict car collisions. However, an important difference is that our predictions are based on both visual cues, proprioceptive sensors, as well as next robot actions.

III. Deep Predictive Models of Physical Perturbations

Our goal is to enable an intelligent system to anticipate undesired external perturbations before they occur. Following the biological inspiration, repeated exposure to a sensory cue preceding a physical perturbation will turn the cue into a predictive variable that helps anticipate the perturbation. To this end, we propose the deep predictive model seen in Fig. 2.

The first module of the DPM is a *deep dynamics module* that learns to discriminate between external perturbations caused by outside events and natural variations of sensor readings due to the currently executed behavior and sensor noise. The deep dynamics module is trained to generate a signal, whenever the recorded sensor values cannot be explained by the actions of the robot. This produces a dense training signal for self-supervised learning of external perturbations. Prediction in the deep dynamics model is performed within a probabilistic framework in order to estimate the model uncertainties.

The second module of the DPM is the *perceptual anticipation module*– a deep network that takes visual percepts, proprioceptive states, and intended actions as input and generates predictions over future noxious signals. It learns to associate specific visual and proprioceptive cues to harmful states. The perceptual anticipation module contains convolutional recurrent layers [16], [17] that process the visual input in both space and time. In addition, it features recurrent and dense layers, that combine the processed visual information with information about the robot state and actions to produce multiple predictions over expected perturbations. Learning is performed using the paradigm of self-supervised learning. More specifically, the training signal produced by the deep dynamics module is used as a target signal. No human intervention is need in order to either provide new training data or label existing data.

A. Deep Dynamics Model

The task of the deep dynamics model is to identify exogenous perturbations affecting the robot. Detecting such perturbations in the sensor stream can be challenging when the robot is performing dynamic movements that by themselves cause fluctuation in the sensor readings. To discriminate between exogenous and endogenous perturbations, we will use a strategy inspired by the human motor system.

Before sending an action $\mathbf{a}_t \in \mathbb{R}^Q$ to the actuators, the robot creates a copy of $\mathbf{a}_t$, the so-called efference copy, and predicts the expected sensory stimuli after execution. This prediction is performed by the deep dynamics model, a neural network that maps the current state $\mathbf{s}_t$ and intended action $\mathbf{a}_t$ onto the expected next state $\mathbf{s}^*_{t+1}$.

After executing action $\mathbf{a}_t$, we can measure the discrepancy between the expected sensations and the measured sensor values, also called the reafference. The degree of discrepancy is an indicator for external physical perturbations.

This methodology is related to the concept of disturbance observers [6]. However, in contrast to disturbance observers, no explicit model of the system needs to be provided. Instead, the deep dynamics model is entirely learned, which is particularly important for compliant and cable-driven robots, for which analytical models can often be hard to devise and difficult to calibrate.

Data Collection: Without loss of generality, we define for the remainder of the paper the system state to be $\mathbf{s}_t = [\boldsymbol{\theta_t}, \dot{\boldsymbol{\theta}}_{\boldsymbol{t}}, \ddot{\boldsymbol{\theta}}_{\boldsymbol{t}}, \mathbf{p_t}]^T \in \mathbb{R}^P$ which includes joint angles $\boldsymbol{\theta_t}$, joint velocities $\dot{\boldsymbol{\theta}}_{\boldsymbol{t}}$, accelerations $\ddot{\boldsymbol{\theta}}_{\boldsymbol{t}}$ and end-effector pose $\mathbf{p_t}$.

To collect training data for training the deep dynamics model, we use motor babbling [18], [19]. To this end, small changes are applied to the the control action $\mathbf{a}_t \in \mathbb{R}^Q$, where Q is the number of degrees of freedom for the robot, i.e., the number of joints. Considering a naive implementation of motor babbling, the action can be sampled from an isotropic Gaussian distribution $\mathbf{a}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. However, empirically we have observed that such a naive sampling approach does not effectively cover the state-action space.

To alleviate this problem, we use a bi-modal distribution and incorporate a simple momentum term

$$\begin{aligned} &\pi \sim \mathcal{B}(0.5) \\ &\mathbf{u} \sim \pi\ \mathcal{N}(\mu\mathbf{1}, \sigma^2\mathbf{I}) + (1-\pi)\ \mathcal{N}(-\mu\mathbf{1}, \sigma^2\mathbf{I}) \\ &\mathbf{a}_t = (\mathbf{u} + \mathbf{a}_{t-1})/2 \end{aligned} \tag{1}$$

Actions sampled using the above strategy effectively cover the state-action space and generate trajectories without causing wear and tear. The result of the motor babbling phase, is a dataset for training which consists of N triplets $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ consisting of the current state, current action, and the next state. The individual matrices storing all states and actions are denoted by $\mathbf{S}_t \in \mathbb{R}^{N\times P}$, $\mathbf{A}_t \in \mathbb{R}^{N\times Q}$ and $\mathbf{S}_{t+1} \in \mathbb{R}^{N\times P}$.

Model Learning: The deep dynamics model is an artificial neural network that maps a state $\mathbf{s}_t$ and action $\mathbf{a}_t$ on to the expected next state $\mathbf{s}^*_{t+1}$. Training is performed using Dropout [20] and data collected in the motor babbling process. A Euclidean loss function is used to identify the error.

The neural network generates a point estimate for any set of inputs. However, due to the non-determinism and noise underlying such tasks, it is important to reason about uncertainties when making predictions. To this end, we leverage recent theoretical insights in order to generate probabilistic predictions from a trained neural network. In particular, it was shown in [21] that neural network learning using the Dropout method [20] is equivalent to a Bayesian approximation of a Gaussian Process modeling the training data. Following this insight, we generate a set of predictions $\{\widehat{\mathbf{s}}_1,\ldots, \widehat{\mathbf{s}}_T\}$ from a trained network through T stochastic forward passes. The generated predictions form a possibly complex distribution represented as a population of solutions. We then extract an approximate parametric form of the

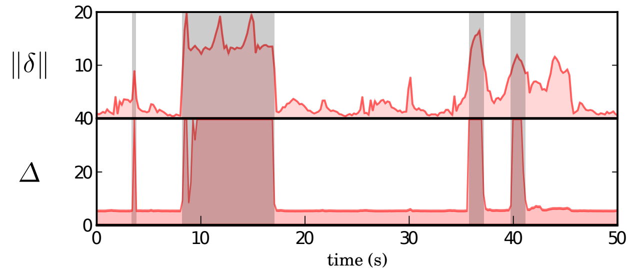


Fig. 3: Analysis of predictive error functions

underlying distribution through moment matching

$$\mathbb{E}(\mathbf{s}^*_{t+1}) \approx \frac{1}{T}\sum_{i=1}^{T}\widehat{\mathbf{s}}_i$$
$$\mathrm{Var}(\mathbf{s}^*_{t+1}) \approx \frac{1}{T}\sum_{i=1}^{T}\widehat{\mathbf{s}}_i^T\widehat{\mathbf{s}}_i - \mathbb{E}(\mathbf{s}^*_{t+1})^T\mathbb{E}(\mathbf{s}^*_{t+1})$$

Given the above distribution moments we can reason about the uncertainty underlying the prediction process.

Perturbation as Predictive Error: As described above and depicted in Fig. 2, we generate in every time step a prediction of the next sensory state $\mathbf{s}^*_{t+1}$ based on the current state and action. Consequently, after executing the action $\mathbf{a}_t$, we measure the *true* sensor values of the robot and compare them to the prediction, i.e., $\boldsymbol{\delta} = \mathbb{E}(\mathbf{s}^*_{t+1}) - \mathbf{s}^*_{t+1}$. Taking the norm of vector $\boldsymbol{\delta}$ we get an estimate of the discrepancy between robot expectation and reality. Assuming a reasonably accurate model, this discrepancy is an estimate of exogenous perturbations that affected the system dynamics. To correct for the inherent model uncertainty and the probabilistic nature of the predictions, we can take an exponential according to $\Delta = \|\boldsymbol{\tau}\|$, with $\tau_i = \exp\big(\delta_i - 2\ \mathrm{Var}(\mathbf{s}^*_{t+1})\big)$.

Fig. 3 depicts the effect the exponential scaling at confidence bound. The ground-truth highlighted in gray was hand-labeled using a video stream as reference. We can see both predictive error functions produce elevated responses during moments of contact. Yet, by incorporating an exponential scaling as performed in Δ we can reduce spurious activations and false-positives.

B. *Perceptual Anticipation Model*

In this section, we will describe how experienced perturbations can be correlated to predictive visual cues. In turn, these visual cues can later be used to anticipate the occurrence of physical contact. For example, perceiving an approaching wall may indicate an impending collision. Still, whether or not an external perturbation will occur is also dependent on the next actions of the robot, e.g., whether or not the robot will stop its course. Hence, states and actions need to be included in the prediction process.

In our approach, a dedicated model – the Perceptual Anticipation Model – learns to predict impending perturbations from a sequence of visual input data, robot actions, and sensory states. Visual input representing the environment is given by an $R \times C$ grid. Each cell of the grid stores F measurements, i.e., depth or color channels that may vary over time. Thus, the visual input corresponds to a tensor $\mathbf{X} \in \mathbb{R}^{R\times C\times N}$.

Training is based on repeated physical interaction with the enviroment,e.g., a human or static objects. Throughout this process, a stream of visuals is recorded using either a traditional RGB camera or an RGB-D depth camera. The result is a sequence of observations $\mathbf{X}_t$. As the same time, states $\mathbf{s}_t$ and actions $\mathbf{a}_t$ of the robot are recorded. In addition, at every time step, the previously introduced deep dynamics model is run, in order to generate estimates p_t of perturbations currently acting on the robot.

Given the above data sets, the goal of training the Perceptual Anticipation Model is to approximate the distribution

$$\begin{aligned}P(p_{t+1},\dots,p_{t+k} \mid\, &\mathbf{X}_{t-j},\dots,\mathbf{X}_t,\\ &\mathbf{s}_{t-j},\dots,\mathbf{s}_t,\\ &\mathbf{a}_{t-j},\dots,\mathbf{a}_t)\end{aligned} \tag{2}$$

where j defines a window of past time steps. More specifically, we are interested in a predictive model that *generates expected future perturbations* $p_{t+1},\cdots,p_{t+k}$ conditioned on *past inputs* $\mathbf{X}_{t-j}$, as well as current and previous robot states and actions.

The presented task requires both attention to spatial patterns within the visual input, as well as attention to temporal patterns and behavioral dynamics. Hence, special care has to be taken to ensure that the model architecture used for learning can effectively identify and incorporate both sources of information when making a prediction.

Network Architecture: In order to base all predictions on both spatial and temporal information, we employ a *deep convolutional recurrent network* to learn the anticipation model. The input of the network is a sequence of visual data, the robot states, and actions. The output of the network is a vector $\mathbf{p} = [p_{t+1},\cdots,p_{t+k}]^T \in \mathbb{R}^k$ that describes the likelihood of a perturbation at each of the time steps $\{t+1,\cdots,t+k\}$. In order to incorporate temporal dynamics, recurrent units are used according to either the Long Short Term Memory (LSTM) [22] model or the Gated Recurrent Units (GRU) [23] model. These recurrent units keep track of a hidden state. In turn, the next state of the network is calculated as a function of the hidden state and the new inputs. In the case of using convLSTM units [24], the network output is governed by the following set of equations

$$\begin{aligned}\mathbf{I}_t &= \sigma(\mathbf{W}_{xi} * \mathbf{X}_t + \mathbf{W}_{hi} * \mathbf{H}_{t-1} + \mathbf{b}_i)\\ \mathbf{J}_t &= \tanh(\mathbf{W}_{xc} * \mathbf{X}_t + \mathbf{W}_{hc} * \mathbf{H}_{t-1} + \mathbf{b}_c)\\ \mathbf{F}_t &= \sigma(\mathbf{W}_{xf} * \mathbf{X}_t + \mathbf{W}_{hf} * \mathbf{H}_{t-1} + \mathbf{b}_f)\\ \mathbf{O}_t &= \sigma(\mathbf{W}_{xo} * \mathbf{X}_t + \mathbf{W}_{ho} * \mathbf{H}_{t-1} + \mathbf{b}_o)\\ \mathbf{C}_t &= \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \mathbf{J}_t\\ \mathbf{H}_t &= \mathbf{O}_t \odot \tanh(\mathbf{C}_t)\end{aligned}$$

where $\mathbf{X}_t$ is the input at time t, $\mathbf{C}_t$ is the memory cell output, and $\mathbf{H}_t$ is the hidden state. Using memory cells is a critical element of LSTM and ensures a that the network output is conditioned on previous activations of the network. The gate variables $\mathbf{I}_t$, $\mathbf{F}_t$, $\mathbf{O}_t$ of the ConvLSTM model denote 3D tensors whose last two dimensions are spatial dimensions. σ denotes the sigmoid function, $*$ is convolution operation and $\odot$ is Hadamard multiplication. In the case of using convGRU units [25], the neural network outputs are defined by equations

$$\begin{aligned}
\mathbf{Z}_t &= \sigma(\mathbf{W}_{xz} * \mathbf{X}_t + \mathbf{W}_{hz} * \mathbf{H}_{t-1} + \mathbf{b}_z) \\
\mathbf{R}_t &= \sigma(\mathbf{W}_{xr} * \mathbf{X}_t + \mathbf{W}_{hr} * \mathbf{H}_{t-1} + \mathbf{b}_r) \\
\widehat{\mathbf{H}}_t &= \Phi(\mathbf{W}_x * \mathbf{X}_t + \mathbf{W}_h * (\mathbf{R}_t \odot \mathbf{H}_{t-1}) + \mathbf{b}) \\
\mathbf{H}_t &= (1 - \mathbf{Z}_t) \odot \mathbf{H}_{t-1} + \mathbf{Z}_t \odot \widehat{\mathbf{H}}_t
\end{aligned}$$

with inputs $\mathbf{X}_1,\ldots,\mathbf{X}_t$, cell outputs/hidden states $\mathbf{H}_1,\ldots,\mathbf{H}_t$ and gates $\mathbf{Z}_t$, $\mathbf{R}_t$ of ConvGRU are 3D tensors with 2 dimensions as the spatial dimensions and one dimension for the convolution filter dimension. The symbol Φ described the activation function used, e.g., $\tanh$ or Rectified Linear Unit (ReLU) [26]. Note that the equations governing the convolutional GRU have fewer parameters, which reduced both training and execution time. Faster forward passes can be crucial for real-time robotics application as the one described here.

Loss Function: In order to train all network parameters, we use a weighted binary crossentropy as a loss function

$$\mathcal{L} = -\frac{1}{T}\sum_{t=1}^{T}\Big(w\mathbf{p}_t \log(\widehat{\mathbf{p}}_t) + (1-\mathbf{p}_t)\left(1-\log(\widehat{\mathbf{p}}_t)\right)\Big) \quad (3)$$

where w is weight penalty, $\mathbf{p}_t$ is the ground truth, and $\widehat{\mathbf{p}}_\mathbf{t}$ is the likelihood of perturbation generated by the network.

IV. Experimental Results

We conducted a human-robot interaction study to investigat the validity of the introduced approach. In all of the following experiments the prediction rate was 5Hz. The parameters were set to: $P = 28, Q = 7, k = 10, w = 3, j = 9, \mu = 0.1$, standard deviation for action sampling $\sigma = 0.05$, dimension of environment data: $row, col = 64$ and channel=1 (gray-scale frame data or depth data).

A. Interaction Dataset

For training the anticipation model, we have created an interaction dataset involving 10 participants. Each participant interacted with the robot for about ten minutes. This resulted in a data set of 3000 data points per person. The participants were instructed to move towards the robot and touch its arm at any location. Throughout this process, the arm of the robot was continously performing a forward-backward movement.

In addition, in 40-50% of the interactions, the participants were instructed to pretend approaching the robot and then turning back without any contact. Incoporating such *feining moves* on the part of the human interaction partner, as well as a constant movement of the arm, ensures that the robot has to constantly monitor the its state and the environment in order to appropriately update its belief about impending physical contact. Data recorded from 9 of the participants was used for training, while 1 participant was used for validation. Data from a separate set of 3 participants was used as test data to evaluate the generalization capabilities of the model.

Two different versions of the training set were generated. First, we created a setup in which depth images were used as input. In the second setup, we used grayscale video images as input.

B. Example Interaction

Fig. 4a shows an example interaction between the robot and a human subject applying forces on its arm. The perturbation identified by the Deep Dynamics Model accurately reflects the moment of contact between the human and the robot. These perturbations are used to train the anticipation model, so that in subsequent interactions the robot can predict the occurence of a perturbation by only observing the human approaching and lifting his hand.

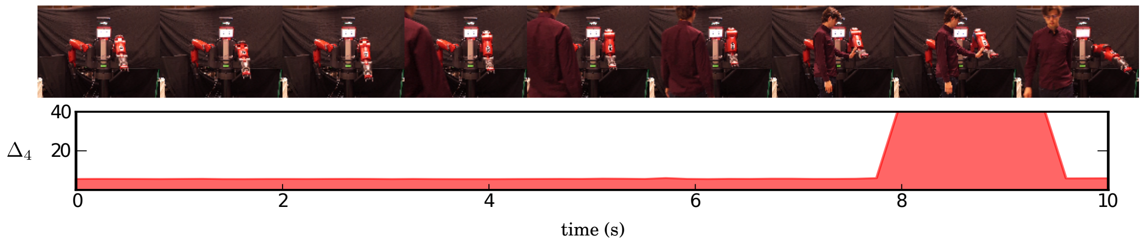
This process can be seen in Fig. 4b. After training the perceptual anticipation model, we provide current observations in form of depth images as input. In turn, the network generates estimates for the next k time steps which reflect the likelihood of a future perturbation at $t + k$. The moment of contact is depicted in Fig. 4b by a vertical line. We can see that the predictions for p_{t+1}, p_{t+5} and p_{t+10} are aligned along a diagonal. The predictor with a larger horizon, i.e., looking ten time steps into the future, activates early to indicate a high likelihood of a contact. The predictors with a shorter horizon activate at later time steps, based ont the horizon they have been trained on. Note the attenuation of the activations, once the robot is outside the envelope for which the predictors have been trained to fire.

Fig. 4c shows the same process with a perceptual anticipation model trained on typical grayscale video images. Again, the predictors fire mostly within the envelope (gray) imposed by the respective horizons, with p_{t+10} firing first. However, the responses are less accentuated when compared with the activations generated from depth images.

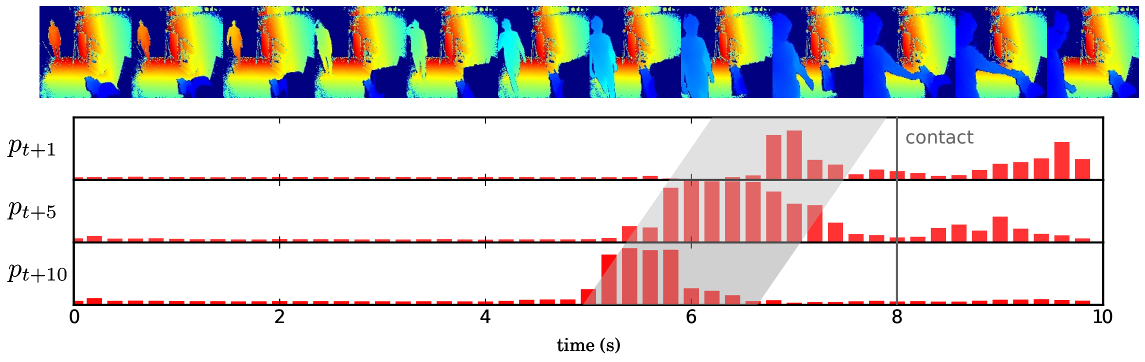
C. Visualizing Network Saliency

To better understand the decision process by which the perceptual anticipation model generates predictions, we visualized the underlying saliency using the method introduced in [27]. Fig. 5 shows an example of saliency maps at different moments in time during a human-robot interaction. Regions highlighted in red or yellow correspond to pixels with a strong influence on the output of the network. Originally, the network is not focusing on any particular region within the image. However, as the human subject starts walking in direction of the robot, the network approximately starts focusing on pixels around the body of both the human subject, as well as the right arm of the robot. As the human comes closer, the saliency scores for each pixel become larger, as indicated by the yellow coloring. At the onset of a contact, the saliency scores attenuate.

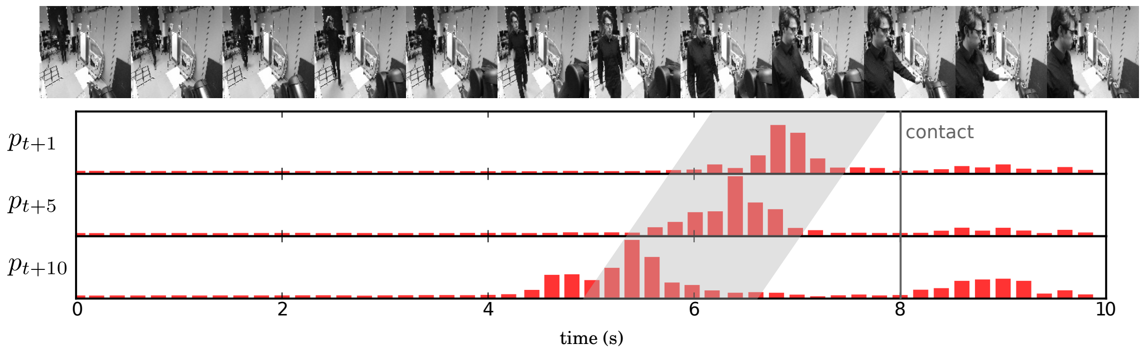
The latter analysis of saliency, is in line with our expectations of features relevant to the task, i.e., the shape and location of the human and robot arm. It is important to note that the activations for this particular predictor (p_{t+5})



(a) Detection of physical perturbations using the Deep Dynamics Model. A human subject approaches the robot, pushes the arm, and moves back. Detected perturbations correspond to the moment of contact and release.



(b) Likelihoods of perturbation predicted by the Perceptual Anticipation Model after self-supervised training. The different predictors fire at different moments ahead of the actual physical perturbation (indicated here by a gray envelope).



(c) Pedictions of the likelihood of physical perturbation generated from a Perceptual Anticipation Model which was trained on grayscale video input. The predictors fire at different moments ahead of the actual moment of contact.

Fig. 4: Interaction Example showing the data and outputs of the Deep Dynamics Model and Perceptual Anticipation Model

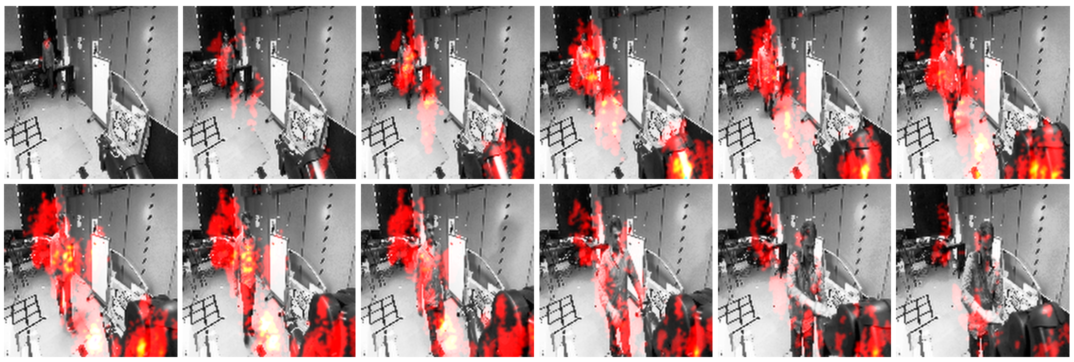


Fig. 5: Saliency map visualization of p_{t+5} predictor. It anticipates any interaction 1s into the future. The images show, highlighted in red, the pixels the network is paying attention when generating an output. In the beginning, almost no region is active, but as the human starts approaching the robot, the network focuses on the human and also on the robot arm. Once the subject is in contact with the robot, the activations attenuates.

quickly attenuate as the human comes close to the robot. We noticed that this pattern is dependent on the specific predictor. Consequently, the saliency maps for the p_{t+1} attenuate at a later moment, while the saliency maps for p_{t+10} attenuate at an earlier moment.

D. Prediction Error on Training and Test Sets

The prediction error on the training set can be seen in Tab. I. Different models were tested, namely (a) ConvLSTM vs. ConvGru, (b) state-action conditioned networks versus convolution only, and (c) depth image vs. frame image.

Config	true+	pred+	precision	recall	MCC
ConvLSTM-SA-D	1907	4607	0.4139	0.6421	0.4953
ConvLSTM-D	2087	5999	0.3479	0.7027	0.4711
ConvGRU-D	1686	4060	0.4153	0.5677	0.4651
ConvGRU-SA-D	1876	5206	0.3604	0.6316	0.4541
ConvGRU-F	2129	6696	0.3180	0.7168	0.4522
ConvGRU-SA-F	2040	6360	0.3208	0.6869	0.4442
ConvLSTM-SA-F	1993	6190	0.3220	0.6710	0.4396
ConvLSTM-F	2127	8039	0.2646	0.7162	0.4060

TABLE I: Comparison between different implementations of the perceptual anticipation model – 2970 perturbation points in 9000.

E. Test Error after Model Selection

Given the above results, we selected a state-action conditioned ConvLSTM (ConvLSTM-SA) as the underlying network model for the prediction. Consequently, we analyzed the predction errors for the generated predictions $p_{t+1}, \cdots, p_{t+k}$ separately. Tab. II and Tab. III show the results of this analysis for depth data and frame data respectively. In both cases precision, recall and MCC (Matthews correlation coefficient) deteriorate, as the network produces predictions for longer horizons. However, using depth input generates significantly better results for short horizon predictors.

pred no	true+	pred+	precision	recall	MCC
p_{t+1}	239	526	0.4544	0.8047	0.5877
p_{t+2}	227	487	0.4661	0.7643	0.5799
p_{t+3}	233	522	0.4464	0.7845	0.5742
p_{t+4}	224	531	0.4218	0.7542	0.5451
p_{t+5}	215	481	0.4470	0.7239	0.5507
p_{t+6}	192	424	0.4528	0.6465	0.5226
p_{t+7}	212	576	0.3681	0.7138	0.4905
p_{t+8}	166	443	0.3747	0.5589	0.4352
p_{t+9}	187	647	0.2890	0.6296	0.3989
p_{t+10}	194	790	0.2456	0.6532	0.3691
ALL	1907	4607	0.4139	0.6421	0.4953

TABLE II: ConvLSTM-SA Depth test (297 perturbation points in 9000)

The above tables show a relatively low precision on the test set. To investigate this phenomenon, we visualized the ground truth versus the generated predictions by the network. An exerpt of this can be seen in Fig. 6. The figure shows the network activations for two different subjects over time. The black marks indicate the moments of contact, i.e., the ground truth (GT). The ten plots in red underneath the ground truth indicate the output predictions of the network. We can see that in the majority of cases the network performs the right classification. We can also see that the activations are aligned along a diagonal, since their outputs fire for different horizons. In some cases (Subject 2, time step 18) the earlier predictors start to fire but immediately cease thereafter. An analysis of the video showed that these cases correspond to the feigning moves the subjects were asked to perform from time to time. This shows that the network performs according to our expectations: the early predictors fire, while the late predictors are awaiting more evidence. Hence, the low accuracy is mostly caused by the network sometimes firing slghtly ahead of time.

pred no	true+	pred+	precision	recall	MCC
p_{t+1}	202	492	0.4106	0.6801	0.5083
p_{t+2}	208	539	0.3859	0.7003	0.4986
p_{t+3}	203	530	0.3830	0.6835	0.4901
p_{t+4}	196	516	0.3798	0.6599	0.4788
p_{t+5}	182	486	0.3745	0.6128	0.4567
p_{t+6}	193	545	0.3541	0.6498	0.4564
p_{t+7}	209	641	0.3261	0.7037	0.4543
p_{t+8}	171	493	0.3469	0.5758	0.4230
p_{t+9}	174	535	0.3252	0.5859	0.4113
p_{t+10}	185	755	0.2450	0.6229	0.3592
ALL	1993	6190	0.3220	0.6710	0.4396

TABLE III: ConvLSTM-SA Frame test (297/9000)

To get a better estimate of the overall prediction accuracy we performed another evaluation in which we counted the number of times all ten predictors activate ahead of a ground truth activation for different participants, see Fig. 7. We can see that the networks performed well for a two out of three subjects, i.e., accuracy of 90% using depth images. For participant P3, accuracies dropped to about 75% (depth) and 60% (grayscale frame data).

V. CONCLUSIONS

In this paper, we introduced a novel methodology that allows robots to associate perceptual cues with impending physical impact to the body. By learning a mapping between observed visual features and future perturbations, robots can anticipate upcoming hazardous or unwanted states. To this end, we introduced a complex neural network model that combines spatial, temporal, and probabilistic reasoning in order to generate predictions over a horizon of next time steps. The network learns to focus on visual features that are most indicative of future exogenous perturbations.

For future work, we aim at extending the framework, such that the predictions can be conditioned on entire control policies of the robot and not only a single next action. Furthermore, we will incorporate prediction framework into reinforcement learning, in order to also autonomously learn preventive motions for avoiding perturbations.

REFERENCES

[1] I. Asimov, *I, Robot*. Fawcett Publications, 1950.

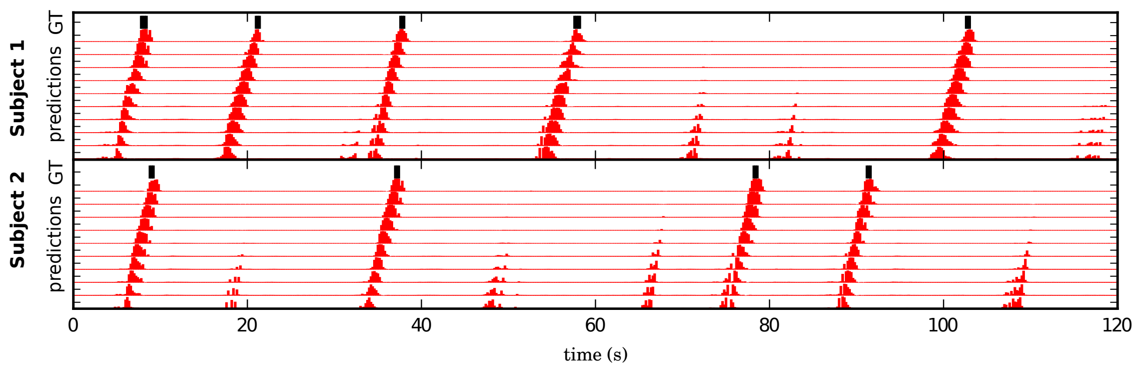


Fig. 6: Visualization of the ground truth timing (black) of a physical perturbation and the activations (red) of the anticipation model for two test subjects.

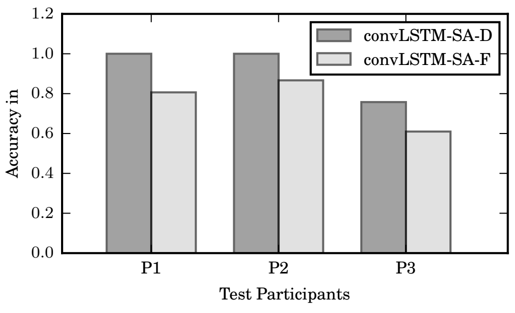


Fig. 7: Percentage of times all ten predictors activate ahead of ground truth activation.

[2] S. Haddadin, *Towards Safe Robots: Approaching Asimov's 1st Law*. Springer Publishing Company, Incorporated, 2013.

[3] A. De Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the dlr-iii lightweight manipulator arm," in *International Conference on Intelligent Robots and Systems (IROS), 2006 IEEE/RSJ*. IEEE, 2006, pp. 1623–1630.

[4] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3356–3363.

[5] H. Iwata, H. Hoshino, T. Morita, and S. Sugano, "Force detectable surface covers for humanoid robots," in *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings (Cat. No.01TH8556)*, vol. 2, 2001, pp. 1205–1210 vol.2.

[6] K. S. Eom, I. H. Suh, W. K. Chung, and S. R. Oh, "Disturbance observer based force control of robot manipulator without force sensor," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, vol. 4, May 1998, pp. 3012–3017 vol.4.

[7] T. Murakami, F. Yu, and K. Ohnishi, "Torque sensorless control in multidegree-of-freedom manipulator," *IEEE Transactions on Industrial Electronics*, vol. 40, no. 2, pp. 259–265, 1993.

[8] E. Berger, D. Müller, D. Vogt, B. Jung, and H. B. Amor, "Transfer entropy for feature extraction in physical human-robot interaction: Detecting perturbations from low-cost sensors," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 829–834.

[9] E. Berger, M. Sastuba, D. Vogt, B. Jung, and H. B. Amor, "Dynamic mode decomposition for perturbation estimation in human robot interaction," in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2014, pp. 593–600.

[10] E. Berger, M. Sastuba, D. Vogt, B. Jung, and H. Ben Amor, "Estimation of perturbations in robotic behavior using dynamic mode decomposition," *Advanced Robotics*, vol. 29, no. 5, pp. 331–343, 2015.

[11] J. Kuehn and S. Haddadin, "An artificial robot nervous system to teach robots how to feel pain and reflexively react to potentially damaging contacts," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 72–79, 2017.

[12] J. Mainprice, R. Hayne, and D. Berenson, "Goal set inverse optimal control and iterative replanning for predicting human reaching motions in shared workspaces," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 897–908, Aug 2016.

[13] C. Pérez-D'Arpino and J. A. Shah, "Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[14] H. B. Amor, G. Neumann, S. Kamthe, O. Kroemer, and J. Peters, "Interaction primitives for human-robot cooperation tasks," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 2831–2837.

[15] F.-H. Chan, Y.-T. Chen, Y. Xiang, and M. Sun, "Anticipating accidents in dashcam videos," in *Asian Conference Computer Vision (ACCV).*

[16] P. H. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene labeling." in *ICML*, 2014, pp. 82–90.

[17] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[18] R. Saegusa, G. Metta, G. Sandini, and S. Sakka, "Active motor babbling for sensorimotor learning," in *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*. IEEE, 2009, pp. 794–799.

[19] Y. Demiris and A. Dearden, "From motor babbling to hierarchical learning by imitation: a robot developmental pathway," 2005.

[20] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[21] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," *arXiv preprint arXiv:1506.02142*, vol. 2, 2015.

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[24] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems*, 2015, pp. 802–810.

[25] Ł. Kaiser and I. Sutskever, "Neural gpus learn algorithms," *arXiv preprint arXiv:1511.08228*, 2015.

[26] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[27] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.