Reliable Vision-Based Grasping Target Recognition for Upper Limb Prostheses

Boxuan Zhong, He Huang, *Senior Member, IEEE*, and Edgar Lobaton, *Member, IEEE*

***Abstract*—Computer vision has shown promising potential in wearable robotics applications (e.g., human grasping target prediction and context understanding). However, in practice, the performance of computer vision algorithms is challenged by insufficient or biased training, observation noise, cluttered background, etc. By leveraging Bayesian deep learning (BDL), we have developed a novel, reliable vision-based framework to assist upper limb prosthesis grasping during arm reaching. This framework can measure different types of uncertainties from the model and data for grasping target recognition in realistic and challenging scenarios. A probability calibration network was developed to fuse the uncertainty measures into one calibrated probability for online decision making. We formulated the problem as the prediction of grasping target while arm reaching. Specifically, we developed a 3-D simulation platform to simulate and analyze the performance of vision algorithms under several common challenging scenarios in practice. In addition, we integrated our approach into a shared control framework of a prosthetic arm and demonstrated its potential at assisting human participants with fluent target reaching and grasping tasks.**

***Index Terms*—Bayesian deep learning (BDL), grasping strategy, reliable computer vision, upper limb prosthesis.**

I. Introduction

COMMERCIALLY available upper limb prostheses make use of muscle activation signals to control a single degree of freedom of the device and switch between various modes, which makes control of these devices unnatural, inefficient, and mentally taxing. Automation based on visual and inertial sensing can alleviate these issues. Computer vision has demonstrated great potential in wearable robotics applications (e.g., human intent prediction and context understanding) because it is informative and noninterrupting (i.e., it can be used to obtain continuous context awareness that can be used for automation of some of the robotic components). For example, computer vision has been used to acquire orientation, shape, and size parameters of the objects to assist grasping performance for upper limb prostheses [1]. DeGol *et al.* [2] adopted deep learning techniques to classify the objects in images and select the appropriate grasping gestures of a prosthetic hand accordingly. However, tasks in those studies were accomplished with a single object in the field of view, uncluttered background, and limited variability on the physical setup—for example, vision-recognition tasks were usually evaluated in scenarios where the prosthetic hand had to stop in front of the target, producing unnatural, disrupted reaching, and grasping motion. A recent study demonstrated the potential of computer vision for human intent recognition under real-life scenarios [3]. Cameras were mounted on human hands to predict daily activities in uncontrolled environments. These studies did not explore reliable computer vision solutions or demonstrate the feasibility for wearable robotics applications. For wearable robots, mistaken actions can be prevented by estimating the uncertainty in the algorithm's output. For example, if the vision algorithm is uncertain with the decision, the accommodations could be "no action" or letting the users take over the control. Furthermore, for human–machine shared control, uncertainty measures of the algorithm's decision can be used to divide the control between human and machine (arbitration) [4].

Deep learning has shown significant potential in a number of applications, such as object detection [5] and semantic segmentation [6]. However, the performance of the neural networks is challenged by insufficient or biased training, observation noises, cluttered background, etc. Progress has been made to understand the competence of deep learning-based vision algorithms, such as class activation maps [7] and studying adversarial examples [8]. An alternative way of understanding these algorithms is via uncertainty quantification. Unfortunately, most modern deep learning models cannot capture the uncertainty of the predictions well. To address this issue, researchers combined the Bayesian probabilistic framework with neural networks, leading to Bayesian neural networks (BNNs) [9]. More recently, Gal and Ghahramani [10] proposed a simple, but effective, Bayesian approximation framework via standard dropout. This provided the potential to seamlessly combine the advantages of BNN and the state-of-the-art deep neural-network architectures. Kendall and Gal [11] then extended this framework to quantify aleatoric and epistemic uncertainty in a unified model. Epistemic uncertainty (model uncertainty) captures the uncertainty of the model caused by insufficient training data, unseen scenarios,

Manuscript received March 2, 2020; accepted May 18, 2020. This work was supported in part by the National Science Foundation under Award 1552828, Award 1527202, and Award 1856441, and in part by the Department of Defense under Award W81XWH-15-C-0125 and Award W81XWH-15-1-0407. This article was recommended by Associate Editor X. Zhu. *(Corresponding authors: He Huang; Edgar Lobaton.)*

Boxuan Zhong and Edgar Lobaton are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695 USA (e-mail: bzhong2@ncsu.edu; edgar.lobaton@ncsu.edu).

He Huang is with the Joint Department of Biomedical Engineering, North Carolina State University, Raleigh, NC 27695 USA, and also with the Joint Department of Biomedical Engineering, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599 USA (e-mail: hhuang11@ncsu.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2020.2996960

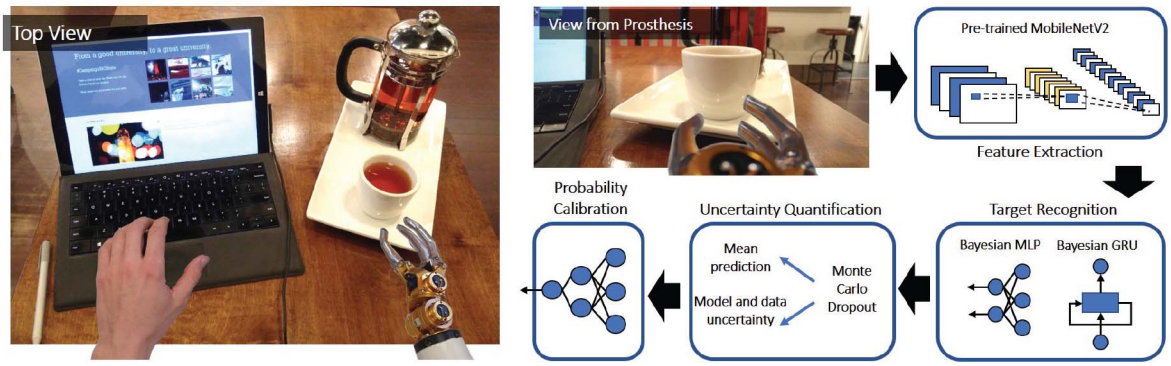

Fig. 1. Reliable vision-based framework for upper limb prosthesis grasping. Images taken from a view corresponding to the forearm are passed through our feature extraction, target recognition, and uncertainty quantification pipelines. Finally, the uncertainty measures are fused together into a single calibrated probability for decision making.

etc. Aleatoric uncertainty (data uncertainty) is data dependent and caused by insufficient sensing modalities or noise in the input. In addition, efforts have been made to extend BNNs to other neural-network architectures, such as convolutional neural networks (CNNs) [12] and recurrent neural networks (RNNs) [13]. The capability of capturing uncertainty provides BNN with great potential in reinforcement learning [14], active learning [15], and safety-sensitive systems, such as autonomous driving [16] and medical applications [17].

Several open questions and challenges still exist for BNNs. First, most of the BNN evaluations have used one of two extremes: 1) synthetic data points sampled from explicit functions or 2) uncontrolled real-life datasets. The former is too simple to represent realistic visual tasks while the later contains multiple uncontrolled factors that influence algorithm behavior. Thus, in real-life settings, the behavior of BNNs has not been fully interpreted. Second, a unified, calibrated, and interpretable confidence estimation is desired to clear the criterion of accepting a prediction in human–machine systems. Third, present BNN applications are limited to traditional classification and regression tasks, such as semantic segmentation and depth estimation [11]. Vision tasks for wearable robots are more challenging due to the high uncertainty in human behavior and actual ambiguities present in these scenarios. The performance of BNNs on vision tasks for wearable robots is unknown.

Human motion during reaching and grasping is fluent. Humans prepare for grasping the targeted object by adjusting hand orientation and aperture angle during the arm reaching phase. Our long-term research goal is to use computer vision and continuous shared control to achieve natural reaching and grasping performance with a prosthesis. Hence, prosthesis grasping, in order to mimic human action, is a complex problem with multiple subtasks, such as detecting grasping target while arm reaching, detecting grasping gestures from other daily activities, and selecting appropriate grasping strategies (e.g., hand gesture, wrist orientation, and contacting point). In this article, we focused on predicting grasping target during arm reaching in challenging scenarios (e.g., multiple objects appeared on the table with a cluttered background). We assume that the grasping action, strategy, and timing (e.g., contact point) have been given because these also depend on the type of end effector of the prosthesis and can be operated by human users. We trained the vision models to identify the target object based on the video streams from the camera mounted on the forearm (Fig. 1) during arm dynamic motions. Our specific contributions are as follows:

1) a new framework for grasping target recognition and uncertainty quantification which incorporates a Bayesian deep neural network that produces three categories of uncertainties and a probability calibration network that projects the three uncalibrated uncertainties into a single calibrated confidence measure;
2) a new methodology for evaluation of this framework using synthetic data and real arm-trajectory data grasping objects in virtual scenes; we demonstrate the effect of several factors, including ambiguous observations, noisy data, new unseen targets and scenes, and additional inertial sensing;
3) a proof-of-concept demonstration of our framework in a real shared control mechanism with humans in the loop in order to measure the impact of uncertainty quantification on the grasping task;
4) a discussion of the scalability of our framework to accommodate real-world situations, complex actuation, other sensing, and the need for training data.

The remainder of this article is organized as follows. In Section II, we describe our reliable vision-based framework followed by the explanation of uncertainty estimation with Bayesian deep learning (BDL) and a probability calibration strategy. In Section III, we explain our procedure of verifying and analyzing the framework, and the corresponding results are shown in Section IV. Afterward, we discuss the extension and generalization potential of our framework. Finally, we conclude this article in Section V by summarizing and offering our insights for future work.

II. Method

A. Problem Statement

In this article, we focus on target recognition during arm reaching for upper limb prosthesis grasping. Note that there are many ways to achieve actual grasping, which is not the

focus of this article. One of the reasons is that there have been excellent publications in robotics that have already partially or successfully addressed the challenge such as [18]–[20]. In addition, the actual grasping action for prostheses depends on the type of end effector (e.g., prosthetic hook versus dexterous hand) and the action that can be achieved easily by the prosthesis user—this means that the grasping does not have to be fully autonomous. One simple solution is predefining the grasping gesture/orientation based on the target. The actual grasping action of the prosthesis can be carried out by the user via EMG signals or other user interfaces (as shown in our real-time demos). In addition, inertial measurement units (IMUs) can be used to determine the grasping task contexts so that the vision framework can focus on the target recognition for prosthesis control. In [21], IMUs were used to detect grasping from general reaching movements, and in [22], the IMU-based approach showed promising results in distinguishing various daily activities, including cleaning a table, typing a document, and carrying a box.

Unlike grasping for robots, a prosthesis does not know the grasping target and the humans can share the control. Considering these facts, this article focused on analyzing the vision algorithms during reaching to predict grasping target and make participatory actions to adjust wrist postural for easy grasping, for which producing continuous and coordinated human-like motions is important. We consider a simple setup which uses a robotic prosthesis (Fig. 1) attached to the forearm of an individual with two degrees of freedom: 1) a gripper manually controlled by the user and 2) a wrist automatically rotated by the machine. The camera and the gripper are rigidly connected except for the rotation at the wrist. For simplicity, we consider a single grasping gesture defined over an interval $[0, T]$, where $t = 0$ is when the gesture is recognized and $t = T$ is when the gripper is about to touch the target object Y. There is also a rotation α_Y associated with the angle needed for grasping that object. In addition, there is a set of rigid coordinate transformations g_t parameterized over time that defines the coordinate frame of the forearm (and the camera) and a set of images captured $I_t : \Omega \to \mathbb{R}^3$, where Ω is the image domain and $\mathbb{R}^3$ represents the color space.

Given that we have a set of possible targets $\{Y_k\}$, our goal is to train a classifier f that gives a prediction $\hat{Y}^{(\tau)} = f(\{I_t\}_{t\leq\tau})$ at time τ based on all prior observations and a probability $p^{(\tau)}$ associated with the likelihood of $\hat{Y}^{\tau}$ being correct [i.e., $\mathbb{P}(\hat{Y}^{(\tau)} = Y)$]. Sections II-C and II-D describe how we obtain these estimates using our network structure. This task goes beyond object recognition because it aims at the detection of a target based on how a human moves his/her forearm for grasping. This means that even if multiple objects are observed in an image, the prior associated with the gesture could make it possible to determine which target is the most likely to be grasped. Also, if only one object is visible and it is too far, we may need to specify low probabilities of correctness because the motion may lead us to some other objects as the forearm obtains closer to the target. Finally, we will make use of these predictions to train a decision process that will determine when to actuate the wrist in order to ensure that we obtain the correct grasping orientation α_Y before we obtain to time τ. Section II-E describes our proof-of-concept setup to illustrate this task.

B. Framework Overview

In our experiments, we defined the vision task as identifying the target object based on the video streams from the camera on the forearm. Fig. 1 presents the four main steps of the framework as follows.

1) *Step One:* MobileNetV2 [23] was used to extract features from images. The network was pretrained with the ILSVRC, a large-scale dataset for image classification [24]. MobileNetV2 was designed for mobile visual recognition, and it is well known for its very low computation complexity and accuracy.
2) *Step Two:* A new neural network with dropout applied to each layer was then trained for target recognition. The dropout layers played three roles: a) a standard technique to prevent overfitting during training; b) a practical Bayesian approximation for neural networks; and c) a way to generate stochastic dropout samples during inference for uncertainty quantification. We considered two versions of neural-network structures (Fig. 1). The first one had three fully connected layers (denoted as *Bayesian multilayer perceptron* or BMLP) and the second one had one *variational* gated recurrent unit (GRU) layer between two fully connected layers (denoted as *Bayesian GRU* or BGRU). BMLP performed predictions only based on the current frame while BGRU integrated the information of historical frames. In practice, BGRU was expected to perform better because of its higher robustness to data noises by combining the information from multiple frames. Evaluating BMLP was worthwhile to understand how the information from individual frames contributed to the predictions. Moreover, datasets of nonsequential images are easier to collect and make up a large proportion of available grasping datasets.
3) *Step Three:* Monte Carlo dropout sampling was used to obtain a number of stochastic samples to obtain uncertainty measures and predictions. Three types of uncertainty were modeled for different analyses: a) predictive entropy uncertainty; b) mutual information (MI) uncertainty; and c) data uncertainty. Predictive entropy captures both epistemic and aleatoric uncertainty while the MI captures the epistemic uncertainty [25].
4) *Step Four:* A probability calibration network was trained to map the three uncertainty measures (from step three) to a single calibrated probability estimation. With the calibrated probability, the prediction decision criteria can be easily customized by considering how much the corresponding application tolerates error.

C. Uncertainty Estimation in Bayesian Deep Learning

BNN extends standard networks by modeling the distributions over the weight parameters. Given x is the input and W is the collection of model parameters, we denote the output of a BNN as $f^W(x)$. With a transformation (if needed), the

prior of W can be assumed as $W \sim \mathcal{N}(0, I)$, a standard multivariate Gaussian distribution. Given a training dataset with $X = \{x_1, \ldots, x_N\}$ as the input, $Y = \{y_1, \ldots, y_N\}$ as the corresponding targets, and N as the number of training observations, the Bayesian inference is used to find the posterior distribution $\mathbb{P}(W|X, Y)$ over the model parameters. The prediction for a new input x^* can be predicted via

$$\mathbb{P}\big(y^*|x^*, X, Y\big) = \int \mathbb{P}\big(y^*|x^*, W\big)\, \mathbb{P}(W|X, Y)\, dW. \tag{1}$$

However, $\mathbb{P}(W|X, Y)$ is analytically intractable for deep neural networks. Several inference techniques are available to approximate the posterior with a simpler distribution $q_\theta(W)$, the parameters of which are denoted as θ. These parameters can be estimated by minimizing the Kullback–Leibler divergence

$$KL\big[q_\theta(W) \| \mathbb{P}(W|X, Y)\big]. \tag{2}$$

Here, we adopted the ideas in [10] to use the Monte Carlo dropout sampling to approximate the prediction and quantify the epistemic uncertainty. We formulated the distributions over BNN parameters as Bernoulli distributions and performed the Bayesian approximation via standard deep neural-network dropout. Then, the KL divergence (2) was minimized by minimizing the cross-entropy loss function with a standard optimizer such as stochastic gradient descent. As a result, it is possible to design BNN with the same network structures as present modern deep neural networks.

In addition, we trained the BNN to estimate the aleatoric uncertainty via a data-dependent observation noise parameter σ [11]. The sampled prediction and aleatoric uncertainty were computed via $[\hat{y}, \hat{\sigma}^2] = f^{\hat{W}}(x)$, where $\hat{W}$ were computed via *stochastic dropout sampling* $\hat{W} \sim q_\theta(W)$. The loss function $\mathcal{L}(\theta)$ (associated with the likelihood) to train BNN for regression was

$$\frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{2}\hat{\sigma}_n^{-2}\big\|y_n - \hat{y}_n\big\|^2 + \frac{1}{2}\log\Big(\hat{\sigma}_n^2\Big)\right] \tag{3}$$

where the first term corresponds to the negative log likelihood and the second term regularizes σ. Since the increase of σ decreases the first term and increases the second term, minimizing $\mathcal{L}(\theta)$ encourages the neural network to assign larger σ to the most challenging samples which are usually the samples that are corrupted by noises. In practice, the network is trained to predict $\log(\sigma^2)$ because it is more numerically stable.

For inference, given that the *stochastic dropout sampling* was performed T times for each sample, the approximated prediction was

$$\mathbb{E}(y) \approx \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t \tag{4}$$

and the uncertainty was formulated as

$$V(y) \approx \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t^2 - \left(\frac{1}{T}\sum_{t=1}^{T}\hat{y}_t\right)^2 + \frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}_t^2 \tag{5}$$

where the first two terms represent the epistemic uncertainty and the last term represents the aleatoric uncertainty.

The intermediate regression aleatoric uncertainty was modeled using a *logit/probit* link so that it could be framed as a classification problem. The sampled latent *logit/probit* vector was then defined as $\hat{z}_k = f^{\hat{W}} + \sigma \cdot \epsilon_k$, where ϵ_k followed a multivariate standard Gaussian distribution. Monte Carlo integration was used to approximate the exact distribution, which yielded the corresponding cross-entropy loss function for a single sample

$$\mathcal{L} = \log\frac{1}{K}\sum_{k=1}^{K}\exp\left[\hat{z}_{k,c} - \log\sum_{c'}\exp\big(\hat{z}_{k,c'}\big)\right] \tag{6}$$

where c represented the ground-truth class index in *logit* vector z and K was the number of Monte Carlo samples during training. During inference, given T stochastic dropout samples, the predicted probability vector is given by

$$p^* \approx \frac{1}{T}\sum_{t=1}^{T}\text{Softmax}\Big[f^{\hat{W}_t}(x^*)\Big]. \tag{7}$$

We adopted two additional measures to model the uncertainty for classification: 1) predictive entropy and 2) MI.

1) *Predictive Entropy Uncertainty [26]:* This metric measures the uncertainty of assigning a label to a sample based on the predicted distribution. This measure is large when the predicted distribution is "flat" over different classes and small when the distribution is "sharp" on one of the classes. We approximated it with Monte Carlo sampling

$$\mathbb{H}\big[y^*|x^*, X, Y\big] = -\sum_{c'}\left[\frac{1}{T}\sum_{t=1}^{T}\mathbb{P}\Big(y^* = c'|x^*, \hat{W}_t\Big)\right] \times \log\left[\frac{1}{T}\sum_{t=1}^{T}\mathbb{P}\Big(y^* = c'|x^*, \hat{W}_t\Big)\right]. \tag{8}$$

2) *Mutual Information Uncertainty:* MI between the prediction y^* and the posterior over W captures the neural network's uncertainty in its output [27]. We also approximated it via Monte Carlo sampling

$$\mathbb{I}\big[y^*, W|x^*, X, Y\big] = \mathbb{H}\big[y^*|x^*, X, Y\big] + \frac{1}{T}\sum_{c',t}\Big[\mathbb{P}\Big(y^* = c'|x^*, \hat{W}_t\Big) \times \log\Big(\mathbb{P}\Big(y^* = c'|x^*, \hat{W}_t\Big)\Big)\Big]. \tag{9}$$

D. Probability Calibration

Intuitively, if the probability of an event is 0.8, then this event should occur 80% of the time. The problem addressed in this article is a supervised multiclass classification problem. Given a labeled dataset $X, Y \in \mathcal{X} \times \mathcal{Y}$, these random variables follow a ground-truth joint distribution $\pi(X, Y)$. Given $\mathcal{H}$ as a classifier which also outputs a probability confidence measure, then $\mathcal{H}(x) = [\hat{y}, \hat{p}]$, where $\hat{y}$ is the class prediction (i.e., the

output of our model f^W) and $\hat{p}$ is the estimated confidence. A sufficient condition for calibration is

$$\mathbb{P}\big(\hat{y} = y|\hat{p} = p\big) = p \quad \forall p \in [0, 1]. \tag{10}$$

However, none of the uncertainty measures in our framework were calibrated. This is actually a common issue for Gaussian processes [28], of which BNNs are closely related to. In addition, since the three uncertainty measures in our framework quantify different sources of uncertainty, decision making should depend on all of them. Thus, we create a recalibration model $\Phi : \mathbb{R}^3 \to [0, 1]$ on top of the BNN classifier f^W such that $\Phi \circ f^W$ generates the desired calibrated probability $\hat{p}$. In our framework, we used a small neural network with three fully connected layers as the calibration network.

E. Uncertainty-Aware Shared Control

We integrated our vision framework into a shared control pipeline [Fig. 2(a)] to leverage human input when the vision algorithm was not confident in its prediction. Fig. 2(b) shows the prosthetic arm we used for evaluation. It had manual control commands and used the video feed as input. The gripper was controlled manually via the user's input while the wrist rotation was controlled manually by the user, or automatically by the vision algorithm. The vision algorithm analyzed the video stream from the on-arm camera and predicted which object the user was grasping. The *wrist control unit* in the diagram was a virtual control logic unit. Manual control of the wrist was always enabled regardless of the calibrated probability—the users could switch to manually control mode if they decided to send control commands. To avoid conflicts and user confusion, sending manual control commands would disable the autonomous control for a period of time (3 s in our experiments). If the calibrated probability was less than a threshold (0.5 in our experiment), the prosthetic arm would keep the original position. Otherwise, the wrist would automatically rotate to a predefined angle associated with the object. Here, we predefined the grasping angle for each object as a proof of concept. With sufficient training data, deep neural networks can predict appropriate grasping strategies for unseen objects. We discuss the extension and generalization ability of our framework in Section IV-H.

The prosthetic arm [Fig. 2(b)] connected to a PC with one 1080Ti GPU received manual control commands from the keyboard and sent the control commands to the servos via a USB servo controller. The video was streamed from the camera attached at the bottom, and the gripper and wrist were controlled by two servos. On the keyboard, the *space* key controlled the gripper while the *arrow* keys controlled the wrist. Pressing the *space* key changed the gripper from "close" to "open" or "open" to "close" based on its current state. Pressing the left/right *arrow* keys continuously rotated the wrist to left/right, respectively. The wrist rotation would stop once the largest angles were reached (the range was $[-90, 90]$ degrees using the direction of gravity as the zero-degree reference direction). In the experiments, our shared control and pure manual control were compared. Pure manual control is identical to shared control that is operating with the probability threshold set to 1 (i.e., all of the automatic control commands are ignored).

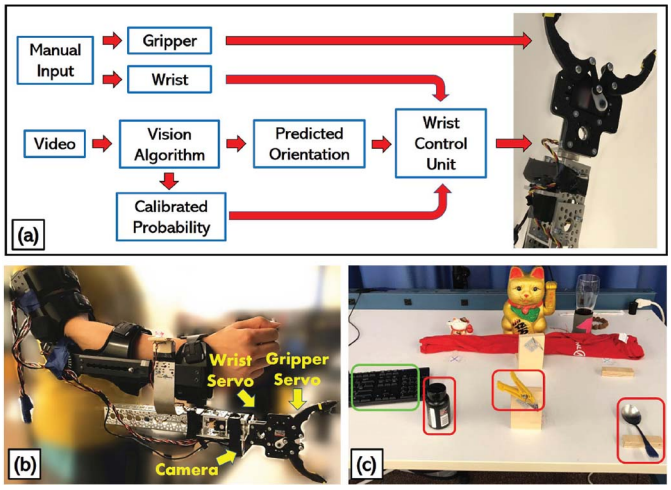

Fig. 2. (a) Diagram of the shared control framework. (b) Prosthetic arm with one camera and two servos for gripper and wrist. (c) Experimental environment setup. The objects in the red boxes are the targets and the keyboard in the green box is for manual control input. All the other objects on the table are used to make the background complex enough to represent a realistic environment.

Fig. 2(c) shows our experimental setup. The objects in the red boxes were the targets to be grasped and manual control commands were performed via the keyboard in the green box. In our experiments, the subjects were asked to grasp and move the objects from one position on the table to another. There are six "*X*" shape signs attached to the table indicating the locations the three objects should be moved to. [Two "*X*" shape signs denoting the start/end positions for an object are shown in 2(c).] The results of this experiment are presented in Section IV-G.

III. Evaluation Methodology

We developed a 3-D simulation environment in Unity3D [29] to evaluate our framework under different situations. In the virtual room scenario (Fig. 3), several objects were placed on a table. Images were captured by simulating the arm-mounted camera's view during reaching and grasping. We assumed that the camera was placed at the center of the forearm, facing the objects. The camera moved together with the arm during grasping. The interpretation of deep neural networks' performance is challenging on uncontrolled real-life datasets since multiple factors can influence the model simultaneously. Thus, we generated two simulation datasets to evaluate our framework: 1) a dataset that used highly controlled synthetic trajectories with designed variations and 2) a dataset that used realistic grasping trajectories captured with human subjects.

A. Synthetic Trajectory Dataset

In this article, we designed segmented trajectories, shown in Fig. 3(d). Segment E was defined as an *evident* segment. Segments A, B, C, D, and F were defined as *ambiguous* segments because the computer vision framework was not

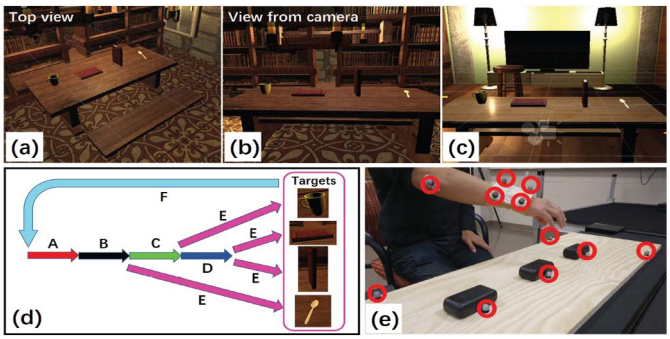

Fig. 3. 3-D simulation setup. (a) Top view of the virtual environment. (b) View from the camera's perspective. (c) Example of a different background. Five different backgrounds were tested. (d) SyTj design. The trajectory followed one of the paths in this graph. Segment A, B, C, and D represent ambiguous segments while approaching the target. Segments E corresponds to the segments where the target should be *evident*. (e) Experimental setup for capturing human grasping trajectories via a motion capture system. The red circles indicate the locations of motion capture markers. The markers on the forearm and elbow were used to calculate the arm's location and orientation while the markers on the table indicate the points that are used as a reference for determining the locations and orientations of the objects.

capable of capturing sufficient information to determine the current grasping intent. When multiple objects are very close to each other or the arm is far from the target, the grasping target recognition task may be ambiguous due to the lack of information from the image captured by the on-arm camera. When the model does not have enough information to determine the current intent, the uncertainty of the model is expected to be high on the *ambiguous* segments. In addition, we simulated four types of noise/artifacts in our experiments: 1) occluded images; 2) task-irrelevant images caused by random camera orientation; 3) motion blur; and 4) illumination variations. Several sample noisy frames can be found in the Appendix [30]. We randomly add this noise to 10% of the training data.

The intent of this dataset was not to generate realistic grasping trajectories; rather, the intent was to emulate scenarios that are typically challenging for grasping tasks. We generated another dataset with realistic human grasping trajectories in the next section.

We simulated four challenging, but commonly occur, scenarios for object grasping. We expected the proposed model to show higher uncertainty under conditions for which some ambiguities, noise, or unknown objects/targets are present.

Scenario One (Baseline): This is a dataset in which no data noise was added, and all targets and background scenes had been previously observed.

Scenario Two (Data Noise): Data noise can severely affect the performance of a computer vision system. We created a testing dataset in which 80% of the data were corrupted by the four types of noise mentioned at the beginning of this section.

Scenario Three (Undefined Targets): We trained the algorithms on four defined targets and studied their behaviors on two undefined targets (UdTg) (i.e., their object class was not part of the training). Sample frames with the UdTg can be found in the Appendix [30].

Scenario Four (Unfamiliar Background): Background variations are usually challenging for computer vision algorithms. One way to solve this problem is to include sufficient variability of the background in the training data. However, this solution is impractical for uncontrolled realistic environments. In this article, we studied the influence of training data variations on our proposed model in handling unfamiliar background (UfBg). Section IV-B specifies how many different backgrounds were used for training and testing.

B. Human Grasping Trajectory Dataset

Real human reaching and grasping motions were recorded by a VICON motion capture system [31] [Fig. 3(e)]. The experimental setup and protocol were approved by the institution review board (IRB) of the University of North Carolina at Chapel Hill. The positions of the markers placed on the forearm and elbow were captured at 100 Hz during arm reaching and grasping.

The experiments included objects which require either horizontal or vertical hand orientations for grasping. In each session, three objects with the same orientation were placed on a table in a row with equal spacing between them. A human subject first sat in front of one object and grasped each object. Each object was repeatedly grasped five to seven times. Then, the subject sat in front of another object and repeated the same grasping tasks. During the experiments, the subjects needed to fully stretch their arms to grasp the furthest objects. In this way, we collected grasping trajectories approached the targets from different directions. This procedure was repeated until the human subjects completed grasping the objects (with the two grasping orientations) at all three sitting locations. Data were collected from two healthy human subjects.

We assumed that a hypothetical camera was placed in the center of the forearm. Therefore, during the data processing phase, we first calculated the camera's positions, angles, and distances to the targeted objects for each frame based on the markers' positions. The full dataset consisted of 108 trials for each object/grasping orientation. Next, we mapped the camera trajectory captured in human experiments to the 3-D simulation space. The scaling factor between the real and the virtual spaces was based on the ratio of the sizes between the real table and the virtual table. Since the dataset contained grasping trajectories from different directions, it was possible to have multiple objects in the camera's field of view including images in which the target object looked smaller than the others. We provided sample frames for this situation in the Appendix [30].

C. Diagnostic Tools for Probability Calibration

In order to determine how good our probability calibration is, we make use of reliability diagrams and standard metrics of their quality.

Reliability Diagrams [32]: They are one of the most common visual tools to evaluate whether a confidence estimation is calibrated. In the diagrams, the true confidence is plotted as a function of empirical probability. For a dataset $\{(x_n, y_n)\}_{n=1}^{N}$ of N realizations of random variables (i.e., our data observations), then $[\hat{y}_n, \hat{p}_n] = \mathcal{H}(x_n)$ are the prediction and corresponding confidence measures for the classifier $\mathcal{H}$. To approximate the true confidence with a finite number of samples, we group

the predictions into K interval bins with the same size $(1/K)$. Given $I_k = ((k-1/K), (k/K)]$ as the kth interval, the number of samples that fall in I_k is $B_k = \sum_{n=1}^{N} \mathbb{1}\{\hat{p}_n \in I_k\}$. Then, the fraction of correctly classified samples

$$\text{acc}_k = \frac{1}{B_k} \sum_{n=1}^{N} \mathbb{1}\left\{\hat{y}_n = y_n,\ \hat{p}_n \in I_k\right\}$$

is an unbiased and consistent estimator of $\mathbb{P}(\hat{y} = \hat{p} | \hat{p} \in I_k)$. Finally, the mean empirical probability for I_k is defined as

$$\text{prob}_k = \frac{1}{B_k} \sum_{\hat{p}_n \in I_k} \hat{p}_n.$$

For a perfect calibrated model, $\text{prob}_k = \text{acc}_k$ for all $k = 1, 2, \ldots, K$.

In addition to reliability diagrams, we also calculated the histograms of the empirical probabilities over all the testing samples because reliability diagrams only show whether a probability interval is calibrated, not how many samples are calibrated (see Fig. 7 for an example).

Quantitative Metrics: Expected calibration error (ECE) and maximum calibration error (MCE) [33] are two statistical measures of miscalibration. For a perfect calibrated probability, both of them equal zero. The ECE is

$$\text{ECE} = \frac{1}{N} \sum_{k=1}^{K} B_k |\, \text{acc}_k - \text{prob}_k \,| \tag{11}$$

and the MCE is

$$\text{MCE} = \max_{k=1}^{K} |\, \text{acc}_k - \text{prob}_k \,|. \tag{12}$$

D. Decision-Making Metrics

We proposed two metrics to evaluate our framework as follows.

1) *Success Rate:* Success rate (SR) measures the predictions' accuracy at the trial level. Each grasping trial began with the arm in a starting position and ended when the arm stopped in front of the object to be grasped. The trial was labeled as successful if the identified targeted object at the end of the trial was correct. The SR was defined to be the fraction of the number of successful trials among all the trials. Hence, the SR was not influenced by the predictions on *ambiguous* segments.
2) *Number of Prediction Changes:* The number of prediction changes (NPC) measures the number of changes of the predicted object orientations along one grasping trajectory. A lower NPC is desired for two reasons: a) a low number of actions reduces the prosthesis' power consumption and b) prosthesis actions that mismatch the user's intent may lead to devise rejection by users.

E. Real-Life Shared Control System Evaluation

The protocol for this evaluation made use of three different objects with grasping angles equal to 0 °, 45 °, and 90 ° [Fig. 2(c)]. The six "*X*" shape signs on the table were placed in two rows and three columns. At the beginning of one session, the three objects were placed on the row closer to the subjects. The subjects were asked to move the objects to the corresponding "*X*" shape signs on the other row and then move them back. While grasping, the subjects were asked to perform a dual cognitive task (counting backward) similarly as the one used in [34]: at the beginning of each session, the subjects were given a random number from 80 to 100 and needed to count backward with a step of 2. We used the dual task to increase the difficulty of the task by adding a cognitive load of the users. Two 20–30 years old able-bodied subjects participated in the experiment (one male and one female). The time spent with our shared control and pure manual control was compared. To eliminate the effect of the order of the objects, each subject conducted 12 sessions for each control with two sets of all six permutations of the locations of the three objects.

IV. Result and Discussion

A. Implementation Details

The algorithms were implemented with Keras and Tensorflow. The inputs of our framework were $224 \times 224 \times 3$ color images. The values of the pixels were scaled to [0, 1] before they were fed into the MobileNetV2 feature extraction network. MobileNetV2 [23] was pretrained by Keras. For both BMLP and BGRU, the first two layers had 128 nodes and used the activation function *ReLU*. *ReLU* has shown promising results in various computer vision applications because it leads to sparsity and can reduce the likelihood of the vanishing gradient issue [24]. The last layer had two parts: 1) a fully connected layer with *Softmax* activation function and 2) a fully connected layer with no activation function and one node to estimate the data uncertainty σ. We also applied l_2 regularization to both the bias and the kernel, with the parameter being set to 10^{-5}. Dropout was applied to each fully connected layer with the dropout probability set to 0.1. For the *variational* GRU layer, both recurrent and input dropout were used with the dropout probability set to 0.1. The probability calibration network was composed of three fully connected layers with 32, 64, and 1 nodes, respectively. We used the *ReLU* activation function for the first two layers and a *sigmoid* activation function for the last layer. The parameters and structures of the neural networks were optimized via trial and error using the validation dataset.

The target recognition and probability calibration networks were trained using a batch size of 64 and a *cross-entropy* loss function. We did the Monte Carlo dropout sampling for 100 repetitions to obtain the approximated predictions and the uncertainty measures. The experiments were performed on a PC with CPU i7-8700K, two NVIDIA 1080Ti GPUs, and 32-GB RAM. With this platform, the inference time for one frame took 14 ms for feature extraction network, 1.7 ms (2.9 ms) for BMLP (BGRU)-based target recognition network, and 0.7 ms for the probability calibration network.

We created a video[1] demonstrating the behavior of our framework under different scenarios.

[1] https://youtu.be/fFZdTbNutFc

B. Training and Testing Procedure

In our framework, two networks are needed to be trained: 1) the target recognition network and 2) the probability calibration network. The feature extraction network was pretrained on the ILSVRC dataset [24].

In total, we developed five 3-D simulation backgrounds: four of them were used as the *training* and *validation* datasets and one was used to test our framework's performance under UfBgs. We generated the data while permuting the locations of the four targets. For each background, we generated four sets of the 24 permutations ($4 \times 24 = 96$ grasping trials per target) of data. We randomly chose 85 of the 96 trials as the *training* dataset and the other 11 as the *validation* dataset. Since there were four backgrounds, there were in total $85 \times 4 = 340$ trials per target in the *training* dataset and $11 \times 4 = 44$ trials per target in the *validation* dataset. Finally, we generated four types of *testing* datasets as follows.

1) *Clean Dataset:* No noisy data and backgrounds appeared in training.
2) *Noisy Dataset:* Noisy data occurred 80% of the time and backgrounds appeared in training.
3) *UfBg Dataset:* No noisy data and the background did not appear in the training dataset.
4) *UdTg Dataset:* No noisy data, backgrounds appeared in training, and two of the targets whose object class was not part of the training set.

Each testing dataset had one set of the 24 permutations of data for each background—UfBg used the one background that did not appear in the training dataset while *Clean*, *Noisy*, and *UdTg* used the other four backgrounds.

The above data generation procedure applied to both the synthetic trajectory (SyTj) and the human grasping trajectory (GrTj) datasets.

Then, we trained the target recognition network with the *training* dataset. The *validation* dataset was fed into the trained network for target recognition and uncertainty measures predictions. If a sample was correctly/mistakenly predicted, it was labeled as "1/0." We used these labeled samples to train the calibration network with the three uncertainty measures as the input and the "1/0" labels as the ground-truth output. The selected loss function was *binary cross-entropy*. In this way, the network was trained to fuse the three uncertainty measures into one calibrated probability estimation. Finally, we evaluated the trained framework on the *testing* datasets.

C. Target Recognition Performance

We note that targets are not well defined at every location in a trajectory. In particular, for the synthetic trajectories, targets are only well defined on the *evident* segments. Hence, recognition accuracy needs to be restricted to these segments as shown in Table I. The *Evident*, *Noisy*, and UfBg labels indicate the results for the *evident* segments of the *Clean*, *Noisy*, and UfBg testing datasets, respectively. Detailed descriptions of these testing datasets can be found in Section IV-B. *GrTj* indicates the results for the GrTj dataset (Section III-B), for which we only used the data when the arm was moving toward the targets. The classification accuracy was not high for all methods because our datasets were designed to be challenging, which highlights the need for a methodology for identifying reliable predictions. Compared to standard neural networks (e.g., MLP and GRU), our framework (BMLP and BGRU) showed better performance because it inherited the advantages of modern deep neural networks and utilized a superior representation of the weight parameters and observations by modeling their corresponding distributions instead of only tracking their deterministic values. For the *Noisy* dataset, BGRU/GRU showed better results than BMLP/MLP because BGRU/GRU was able to integrate the information from multiple corrupted frames. For *Evident*, UfBg, and *GrTj*, BMLP/MLP showed better performance than BGRU/GRU because BGRU/GRU had higher model complexity which could lead to overfitting issues.

TABLE I
TARGET RECOGNITION ACCURACY (%). HIGHEST (**BLUE**) AND SECOND HIGHEST (**GREEN**) ACCURACY ARE HIGHLIGHTED

Dataset Type	BMLP	BGRU	MLP	GRU
Evident	***73.02***	72.37	**73.70**	71.17
Noisy	48.60	**62.97**	49.01	***59.24***
UfBg	**51.96**	44.42	***48.79***	42.72
GrTj	**82.42**	81.42	***81.58***	81.02

D. Uncertainty Quantification

In this section, we analyze the distributions of the uncertainty measures computed for the different scenarios considered. This type of analysis can help identify the sources of uncertainty and their discriminative power between *evident* and *ambiguous* segments.

Uncertainty From Insufficient Training Data: We studied the effect of the amount of training data on our framework with two training dataset sizes: 5 and 340 trials for each target. Fig. 4 shows the results for BMLP on the SyTj dataset. The color codes and letters of the segments matched the ones shown in Fig. 3. We provided the results for Bayesian GRU (BGRU) in the Appendix [30]. The x-axis indicates the frame ID (representing time) and the y-axis indicates the values of the uncertainty measures, calibrated probability, target recognition, and distance between the hypothetical camera and targets.

As shown in Fig. 4(a), with an insufficient amount of training data, MI was more significant than the entropy uncertainty because the neural network was not well trained and hence not confident in its output. As highlighted by the green arrows, with more training data, entropy uncertainty started to play a more significant role because the neural network was more confident in its outputs but not confident in assigning labels because of the ambiguity in the data. The data uncertainty was high with little training data because most of the samples were challenging for such a neural network. With more training data, the data uncertainty started to show similar trends as the entropy uncertainty—high values were assigned to the sample with high data ambiguity.

As highlighted by the red arrows, data and MI uncertainty for the *evident* segments (E) decreased with the increase of the training data size while for the *ambiguous* segments, it

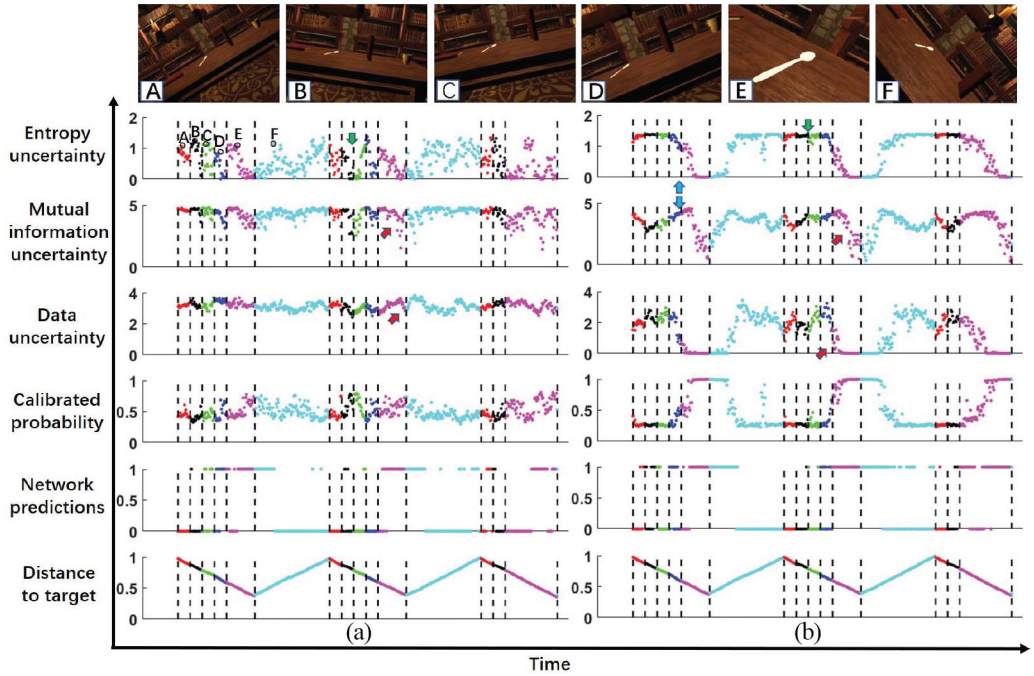

Fig. 4. Uncertainty measures in the SyTj dataset with varying training data sizes: (a) five trials per target and (b) 340 trials per target. The six snapshots on the top are examples of what the camera saw during different segments. The first three rows of the scatter plots are the three measures of uncertainty. The fourth row is the calibrated probability from the probability calibration network. The fifth row is the network's target recognition result, 1 indicates the prediction was correct and 0 indicates an incorrect prediction. Since we assign labels to the entire trajectory, we do not expect the model to make correct predictions for the trajectory segments that are ambiguous (i.e., Segments A, B, C, D, and F). The last row is the distance from the camera to the target. The lower this value is the closer the camera is to the target. The result is based on the *Clean* testing dataset with BMLP as the model.

remained the same. Data uncertainty was not explained away with more training data for the *ambiguous* segments because it captured the essential ambiguity in the data.

As highlighted by the blue arrows, with sufficient training data [Fig. 4(b)], MI and entropy uncertainty showed different trends on the *ambiguous* segments. Entropy uncertainty captures the uncertainty of assigning labels while MI uncertainty captures the network's uncertainty in its output. When the camera approached the target, the BNN model started to assign higher probabilities to some of the classes instead of a "flat" distribution over all the classes, leading to decreased entropy uncertainty. However, the sampled predictions were also getting higher fluctuations since the model was still confused between several of the classes, leading to increased MI uncertainty. This phenomenon was not significant for BGRU (results provided in the Appendix [30]) because BGRU was able to extract temporal features (e.g., the direction of motion) which decreased the entropy uncertainty. In addition, the uncertainties on segment F were expected to have a large range and high fluctuation because we did not include *F* segments in the training of either the target recognition network or the calibration network, and the data generated for this segment had larger variations than the other segments as well.

The above discussion was based on the SyTj dataset. Similar trends were also observed with the GrTj dataset.

Uncertainty Under Challenging Scenarios: We studied the behaviors of BMLP and BGRU under different challenging scenarios. The results of calibrated probability and uncertainty measures are shown in Figs. 5 and 6, respectively. The *y*-axis of each plot represents the percentage of samples with the

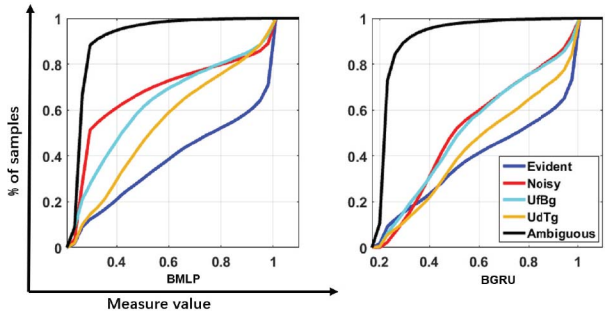

Fig. 5. Sample distribution with respect to the calibrated probability for *Evident*, *Noisy*, UfBg, UdTg, and *Ambiguous* data. Two models are compared: BMLP (left) and BGRU (right). The results are based on sufficient training data (340 trials per target).

calibrated probability or uncertainty values less than the value on the *x*-axis.

Fig. 5 presents the results of the calibrated probability on different testing datasets. *Ambiguous* indicates the results for the *ambiguous* segment of the *Clean* testing dataset. *Evident*, *Noisy*, UfBg, and *UdTg* indicate the results for the *evident* segments of the *Clean*, *Noisy*, UfBg, and UdTg testing datasets, respectively. The results for BMLP (left) and BGRU (right) showed similar trends. The framework was most confident in its predictions on *Evident*, least confident for *Ambiguous*, and the confidences for *Noisy*, UfBg, and *UdTg* were mostly in between as expected. For *Ambiguous*, little information pertinent to the user intent can be captured for target recognition while for *Noisy* and UfBg, it was still possible to obtain enough information for prediction. BGRU was less sensitive

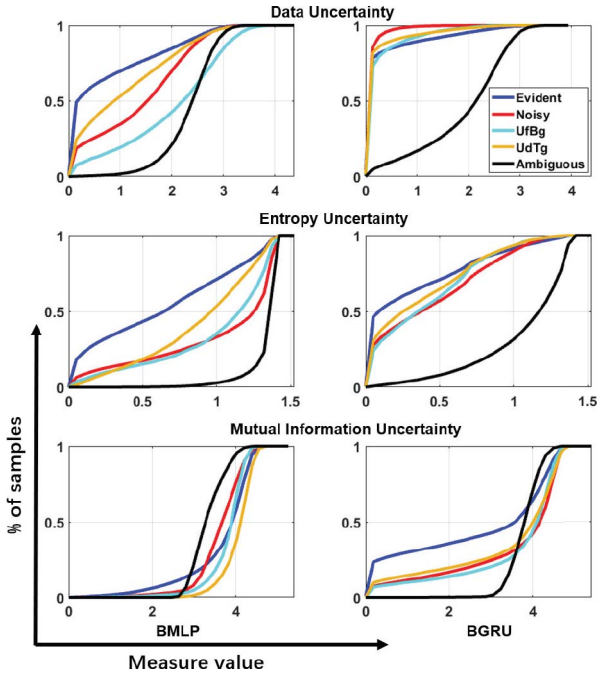

Fig. 6. Sample distribution with respect to different uncertainties for *Evident*, *Noisy*, UfBg, UdTg, and *Ambiguous* data. Two models are involved: BMLP (left) and BGRU (right). The results are based on sufficient training data (340 trials per target).

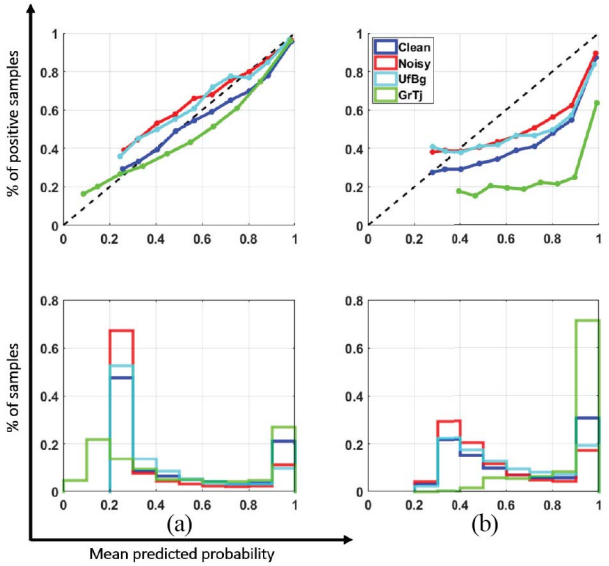

Fig. 7. Confidence histograms (bottom) and reliability diagrams (top) for calibrated probability (left) and *Softmax* probability (right). The results are based on the BMLP model with sufficient training data (340 trials per target).

TABLE II
RESULTS OF ECE AND MCE. LOWEST (**BLUE**) AND SECOND LOWEST (**GREEN**) VALUES ARE HIGHLIGHTED

		Clean	Noisy	UfBg	GrTj	InSf
ECE	BMLP	**0.0330**	0.1077	**0.0934**	**0.0499**	***0.0172***
	BGRU	***0.0605***	**0.0221**	***0.1031***	***0.0552***	**0.0079**
	MLP	0.1511	***0.0919***	0.1326	0.4051	0.3426
	GRU	0.3564	0.4352	0.3838	0.4646	0.4207
MCE	BMLP	**0.1068**	***0.1361***	**0.1168**	***0.1366***	***0.1349***
	BGRU	***0.1295***	**0.0820**	***0.2136***	**0.1167**	**0.0668**
	MLP	0.3351	0.2619	0.3072	0.6453	0.4583
	GRU	0.5712	0.5304	0.4895	0.7043	0.4828

to data noise because it can compensate the disturbance by fusing the information from multiple frames. The uncertainty for *UdTg* was not as high as that for *Ambiguous* because our framework utilized a discriminative model and the uncertainty was measured with respect to the difficulty of classifying the sample into one of the defined classes—for *Ambiguous*, multiple "defined" objects were in the camera's field of view causing higher confusion in classification. In practice, higher uncertainty for *UdTg* is desired which is a limitation for discriminative models.

Fig. 6 shows the behaviors of the three uncertainty measures under different scenarios with BMLP (left) and BGRU (right). For BGRU, *Noisy*, UfBg, and *UdTg* showed low data and entropy uncertainty but high MI uncertainty while BMLP showed the opposite trends. With the capability to combine multiple frames for prediction, BGRU increased its confidence in assigning labels ("sharper" distributions over classes). However, the disturbances decreased the confidence in the outputs of BGRU (higher fluctuation in the sampled predictions). This observation indicates that BGRU had more overfitting issues than BMLP, matching our observations in Section IV-C. *Evident* and *Ambiguous* showed similar distributions of the MI uncertainty for BMLP because the framework yielded higher MI uncertainty at the beginning of the *evident* segment than in the *ambiguous* segments as we observed in Fig. 4.

E. Probability Calibration

This section describes our probability calibration results. A well-calibrated probability measure gives an appropriate measurement of the reliability of a prediction. Fig. 7 shows the reliability diagrams and the histograms of our calibrated probability for BMLP (left) and the *Softmax* probability for standard MLP (right). Detailed descriptions of the testing datasets are in Section IV-B. The results reported in this section were based on the data from both the *ambiguous* and the *evident* segments. Our calibrated probability outperformed *Softmax* probability and was promising in fusing the three uncertainty measures into one calibrated probability. *Softmax* probability tended to assign overestimated probabilities to most of the samples—matching the observations reported in [35]—while our calibrated probability did not show this bias. The results for BGRU/GRU are similar to those for BMLP/MLP and we provided the corresponding results in the Appendix [30].

Table II shows the quantitative measures of model calibration: ECE and MCE. Lower ECE and MCE indicate that the predicted probability is better calibrated. *InSf* indicates the results for the *Clean* testing dataset with a small training dataset—five trials per target. *Clean*, *Noisy*, UfBg, and *GrTj* utilized a large training dataset—340 trials per target. The proposed calibrated probability showed promising results for all scenarios. In addition, MLP was, in general, better calibrated than GRU because of issues with overfitting.

F. Decision-Making Performance

With the calibrated probability, decisions of whether or not to believe the predictions can be easily made by considering how much the application tolerates mistakes. As stated in

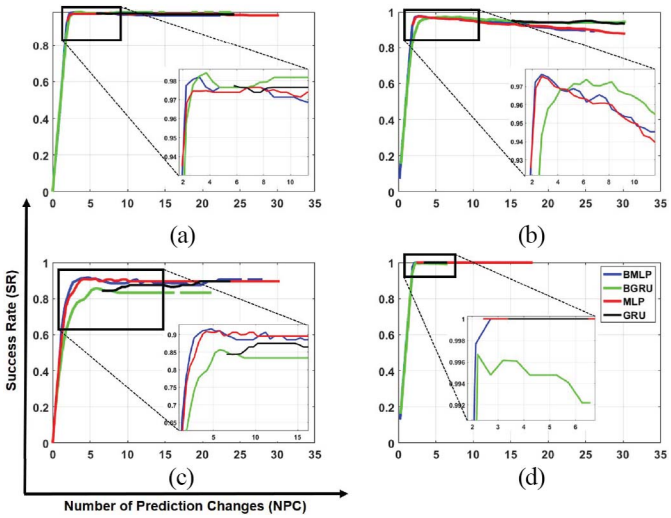

Fig. 8. Plot of SR with respect to NPC. (a) *Clean* dataset with synthetic trajectories. (b) *Noisy* dataset with synthetic trajectories. (c) UfBg dataset with synthetic trajectories. (d) *Clean* dataset with human grasping trajectories (GrTj).

Section III-D, we proposed the use of the SR and NPC to quantify the system's performance. Generally speaking, a good system should have a relatively high SR with a reasonably low NPC. We further proposed to find out a good tradeoff point between SR and NPC based on the following strategy. By changing the probability threshold (θ) for decision making, we obtained different pairs of SR and NPC. The step length of change in θ was 0.002. The results were grouped by different NPC intervals. Here, we took all the results with NPC from 0 to 30 and grouped them into 60 uniform intervals (interval length 0.5). Finally, the averaged NPC and SR for each interval were calculated and plotted in Fig. 8. The "gaps" on the lines indicated that no samples fell in the corresponding intervals in this experiment. In addition, we assumed that the system was always initialized with a default state that was different from any of the targeted states. That is, if no action was taken along the entire trajectory, it was labeled as *incorrect* while calculating SR.

As shown in Fig. 8, MLP and GRU usually could not achieve an NPC lower than 2, even with a high probability threshold (e.g., 0.998) because they were usually overconfident in their predictions—especially for GRU which has higher model complexity. In contrast, BMLP and BGRU showed a better tradeoff curve between SR and NPC and can achieve a high SR with small NPC simultaneously. In Fig. 8(b), when NPC $>$ 4, with the same NPC, BGRU/GRU produced higher SR than BMLP/MLP because BGRU/GRU was able to mitigate the impact of data noises by integrating the information from multiple corrupted frames. For UfBg [Fig. 8(c)], given the same NPC, BMLP/MLP showed better SR than BGRU/GRU because BGRU/GRU required more background variability to avoid overfitting, which matches our observations in Section IV-C.

Compared with the SyTj dataset, the trajectories in the GrTj dataset were more realistic, resulting in grasping intents that were usually clearer. In SyTj, we intentionally designed the *ambiguous* segments for which no clear intent could be observed. Thus, the models show 100% SR on the GrTj dataset with only 2–3 NPC.

TABLE III
AVERAGE TIME SPENT AND STANDARD DEVIATION (SECONDS) WITH OUR SHARED CONTROL AND PURE MANUAL CONTROL FRAMEWORKS

	Shared Control	Manual Control
Subject 1	22.2 (1.6)	25.15 (2.16)
Subject 2	20.03 (1.53)	23.76 (1.01)

G. Performance of Real-Life Shared Control System

Section II-E described our shared control system and Section III-E described our evaluation protocol. In this experiment, we used the same neural-network structure, the number of layers, and the number of units as we described in Section IV-A.

Since each subject conducted 12 sessions, there were 12 data points of recorded time per subject. Table III reports the mean (standard deviation) after deleting the highest and the lowest values for each subject (left with ten data points per subject). Our shared control showed around 14% improvement over the pure manual control in respect of the average time spent for each session. In each session, the subject grasped and placed the objects six times as described in Section III-E. On average, one grasping and placing action took around 3.5 s, which is a reasonable amount of time since the cognitive dual tasks slowed down the grasping action.

H. Extensions and Generalization

This article aimed at developing a reliable vision-based framework to assist upper limb prosthesis grasping. Our analysis focused on the performance of the vision algorithms and the uncertainty measures under controlled but challenging scenarios. In this section, we discuss the extension and generalization capabilities of our framework for real-life upper limb prosthesis grasping.

The framework can be extended to incorporate other sensing modalities, such as depth sensors and IMUs. For example, an IMU can be used to determine the grasping task contexts so that the vision framework could focus on the target recognition for prosthesis control. In this article, we assumed that these techniques were ready and available—in [21], IMU was used to detect grasping from general reaching movements and in [22], the IMU-based approach showed promising results in distinguishing various daily activities, including cleaning up a table, typing a document, and carrying a box. Additional sensing modalities could also be combined with the vision information for better target recognition. As a proof of concept, we incorporated the velocity of the arm into our vision framework and evaluated it on the GrTj dataset. The experiment details and results are in the Appendix [30].

Our target recognition framework can be trained for different tasks based on how the training data are labeled. For example, if the images of the objects are labeled with the appropriate grasps (e.g., pinch, power, etc.), the framework can be trained to decide the prosthetic hand gestures for

grasping [2], [36]. In addition, deep neural networks have shown promising generalization capability for grasping unseen objects [2], [18], [20], [36], [37]. We expect our framework to share similar generalization capabilities to the modern deep neural-network structures that it consists of. We demonstrated this capability by applying our framework to a grasp classification task [2] as shown in the Appendix [30]. DeGol *et al.* [2] achieved 93.2% accuracy for predicting the appropriate grasps for unseen objects whereas our framework (BMLP) achieved 91.2% with a less powerful but more efficient pretrained network.

V. Conclusion

We presented a reliable vision-based framework to assist upper limb prosthesis grasping by recognizing grasping targets in realistic and challenging scenarios during arm reaching. This framework combined the benefits of the Bayesian theory and recent deep learning developments. It was able to quantify the uncertainty of its predictions and estimate a calibrated probability for each prediction. The framework's behaviors under several ambiguous or noisy scenarios were compared and analyzed. Moreover, we showed how to fuse different uncertainty measures into a single calibrated probability that can be applied to decision making. Finally, the algorithm was integrated into a shared control framework that was applied to a prosthetic arm and evaluated using human subjects that performed realistic grasping tasks.

This article focused on evaluating the performance of the vision algorithms and analyzing the uncertainty measures under challenging scenarios. In the future, we plan to incorporate the framework with a more advanced upper limb prosthesis system (e.g., additional sensing modalities or more degrees of freedom) and conduct a comprehensive evaluation in a realistic environment. It is also worthwhile to evaluate and compare the forearm camera design in this article with other data-acquisition designs (e.g., eye-tracking glasses).

Acknowledgment

The authors thank Tianfu Wu, I-Chieh Lee, Turner Richmond, Jeremy Cole, Qian Ge, Szu-yu Lo, Xinran Li, and Jiale Hu for their help and suggestions for this article.

References

[1] M. Markovic, S. Dosen, D. Popovic, B. Graimann, and D. Farina, "Sensor fusion and computer vision for context-aware control of a multi degree-of-freedom prosthesis," *J. Neural Eng.*, vol. 12, no. 6, 2015, Art. no. 066022.

[2] J. DeGol, A. Akhtar, B. Manja, and T. Bretl, "Automatic grasp selection using a camera in a hand prosthesis," in *Proc. IEEE 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, 2016, pp. 431–434.

[3] T.-Y. Wu, T.-A. Chien, C.-S. Chan, C.-W. Hu, and M. Sun, "Anticipating daily intention using on-wrist motion triggered sensing," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 48–56.

[4] D. P. Losey, C. G. McDonald, E. Battaglia, and M. K. O'Malley, "A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction," *Appl. Mech. Rev.*, vol. 70, no. 1, 2018, Art. no. 010804.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.

[7] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2921–2929.

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014. [Online]. Available: arXiv:1412.6572.

[9] R. M. Neal, *Bayesian Learning for Neural Networks*, vol. 118. New York, NY, USA: Springer-Verlag, 1996. [Online]. Available: https://www.springer.com/us/book/9780387947242?gclid=Cj0KCQjwlN32BRCCARIsADZ-J4tEHegZvYdk_VMG8CSS8a_nJfAjQNfkPcctvo1ipd7dIwva8xgMqIcaAnamEALw_wcB

[10] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1050–1059.

[11] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5574–5584.

[12] Y. Gal and Z. Ghahramani, "Bayesian convolutional neural networks with Bernoulli approximate variational inference," 2015. [Online]. Available: arXiv:1506.02158.

[13] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1019–1027.

[14] S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1192–1201.

[15] Y. Gal, R. Islam, and Z. Ghahramani, "Deep Bayesian active learning with image data," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1183–1192.

[16] R. McAllister *et al.*, "Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 4745–4753.

[17] J. van der Westhuizen and J. Lasenby, "Bayesian LSTMs in medicine," 2017. [Online]. Available: arXiv:1706.01242.

[18] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 421–436, 2018.

[19] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2017, pp. 2442–2447.

[20] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2017, pp. 769–776.

[21] J. de Vries, A. van Ommeren, G. Prange-Lasonder, J. Rietman, and P. Veltink, "Detection of the intention to grasp during reach movements," *J. Rehabil. Assistive Technol. Eng.*, vol. 5, pp. 1–9, Jan. 2018.

[22] N. Lokare, S. Samadi, B. Zhong, L. Gonzalez, F. Mohammadzadeh, and E. Lobaton, "Energy-efficient activity recognition via multiple time-scale analysis," in *Proc. IEEE Symp. Series Comput. Intell. (SSCI)*, 2017, pp. 1–7.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[25] L. Smith and Y. Gal, "Understanding measures of uncertainty for adversarial example detection," 2018. [Online]. Available: arXiv:1803.08533.

[26] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.

[27] N. Houlsby, F. Huszár, Z. Ghahramani, and M. Lengyel, "Bayesian active learning for classification and preference learning," 2011. [Online]. Available: arXiv:1112.5745.

[28] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, Dept. Eng., Univ. Cambridge, Cambridge, U.K., 2016.

[29] U. Technologies. (2019). *Unity3D*. [Online]. Available: https://unity3d.com/

[30] (2020). *Reliable Vision-Based Grasping Target Recognition for Upper-Limb Prostheses: Appendix*. [Online]. Available: https://research.ece.ncsu.edu/aros/paper-tcyb2020-upperlimb/

[31] (2019). *Vicon Motion Capture System*. [Online]. Available: https://www.vicon.com/

[32] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proc. ACM 22nd Int. Conf. Mach. Learn.*, 2005, pp. 625–632.
[33] M. P. Naeini, G. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using Bayesian binning," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2901–2907.
[34] S. Pardhan and S. Zuidhoek, "Dual cognitive task affects reaching and grasping behavior in subjects with macular disorders," *Invest. Ophthalmol. Vis. Sci.*, vol. 54, no. 5, pp. 3281–3288, 2013.
[35] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1321–1330.
[36] G. Ghazaei, A. Alameer, P. Degenaar, G. Morgan, and K. Nazarpour, "Deep learning-based artificial vision for grasp classification in myoelectric hands," *J. Neural Eng.*, vol. 14, no. 3, 2017, Art. no. 036025.
[37] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2016, pp. 3406–3413.

Boxuan Zhong (Graduate Student Member, IEEE) received the B.E. degree in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2015, and the Ph.D. degree in electrical and computer engineering from North Carolina State University, Raleigh, NC, USA, in 2020.

His current research interests include computer vision, machine learning, and robotics.

He (Helen) Huang (Senior Member, IEEE) received the Ph.D. degree in biomedical engineering from Arizona State University, Tempe, AZ, USA.

She was a Postdoctoral Fellow of neural engineering with the Rehabilitation Institute of Chicago, Northwestern University, Evanston, IL, USA. She is currently the Jackson Family Distinguished Professor with the Joint Department of Biomedical Engineering, North Carolina State University, Raleigh, NC, USA, and the University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, and the Director for the Closed-Loop Engineering for Advanced Rehabilitation Core. Her research interests lie in neural-machine interfaces for prostheses and exoskeletons, human–robot interaction, adaptive and optimal control of wearable robots, and human movement control.

Dr. Huang was a recipient of the Delsys Prize for Innovation in Electromyography, the Mary E. Switzer Fellowship with NIDRR, and the NSF CAREER Award. Her research has been sponsored by NSF, NIH, DOD, NIDRR, and DARPA. She is a member of the Society for Neuroscience, AAAS, and BMES.

Edgar Lobaton (Member, IEEE) received the Ph.D. degree in electrical engineering and computer sciences from the University of California at Berkeley, Berkeley, CA, USA, in 2009.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA. He was engaged in research with Alcatel-Lucent Bell Labs, Murray Hill, NJ, USA, from 2005 to 2009. He conducted research with the Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, from 2009 to 2011. His research interests include pattern recognition, computer vision, and robotics.

Dr. Lobaton was awarded the 2009 Computer Innovation Fellows Postdoctoral Fellowship.