Sensor Substitution for Video-based Action Recognition

Christian Rupprecht*,1,2, Colin Lea*,2
Federico Tombari[1,3], Nassir Navab[1,2], Gregory D. Hager[2]

***Abstract*— There are many applications where domain-specific sensing, such as accelerometers, kinematics, or force sensing, provide unique and important information for control or for analysis of motion. However, it is not always the case that these sensors can be deployed or accessed beyond laboratory environments. For example, it is possible to instrument humans or robots to measure motion in the laboratory in ways that it is not possible to replicate in the wild. An alternative, which we explore in this paper, is to address situations where accurate sensing is available while training an algorithm, but for which only video is available for deployment. We present two examples of this sensory substitution methodology. The first variation trains a convolutional neural network to regress real-valued signals, including robot end-effector pose, from video. The second example regresses binary signals derived from accelerometer data which signifies when specific objects are in motion. We evaluate these on the JIGSAWS dataset for robotic surgery training assessment and the 50 Salads dataset for modeling complex structured cooking tasks. We evaluate the trained models for video-based action recognition and show that the trained models provide information that is comparable to the sensory signals they replace.**

I. Introduction

Domain-specific sensors are often critically important in robotic applications. For example, pressure sensors are important for manipulation tasks, tool tracking supports visual servoing, and accelerometers support SLAM approaches. However, there are many applications where the ideal sensing is too impractical or too costly to deploy in real-world settings. For instance, in the next-generation kitchen[1] it might be beneficial to attach motion sensors to all tools to support monitoring or control, but retrofitting every kitchen at scale is simply impractical due to a combination of cost, data acquisition and synchronization overhead, and physical constraints in instrumentation. A practical alternative would be to mount a video camera to observe the scene, but current methods for video tracking and action recognition generally achieve worse performance than their counterparts using domain-specific sensing [1], [2].

In this paper, we introduce a methodology and associated models capable of replicating domain-specific sensing from video footage. We use state-of-the-art machine learning techniques to regress a mapping from a video frame to the corresponding sensor measurements in a supervised setting, thus enabling augmentation of videos with *virtual* sensor data whenever real sensors are not available. Empirically we find that this maintains or even improves performance for a task like action recognition. In this work, we demonstrate such ability within the domains of robotic surgery and cooking.

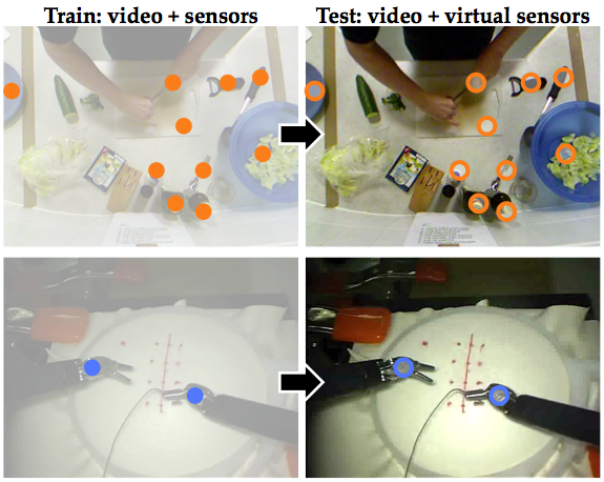

Fig. 1: Our model learns a video-based representation of domain-specific sensors. **(top)** Orange circles correspond to accelerometers placed on objects in the University of Dundee 50 Salads dataset. **(bottom)** Blue circles correspond to end effector positions from a da Vinci robot in the JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS). We train on video and sensor data but test on only video by using sensor data predicted from the video.

In the first domain, we regress position measurements by learning a continuous model for the 3D position of two robot end effectors from a da Vinci surgical robot. We evaluate on the JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS) [3] but similar approaches could be applied to other continuous signals from a mobile robot, a robotic arm, or motion capture.

For applications in other structured activities, i.e. for a cooking assistant, we learn a discrete model to identify when specific objects in a scene are in motion. This is evaluated using the University of Dundee 50 Salads dataset [4] where signals are derived from an accelerometer attached to each cooking utensil. We define each object as `in motion` or `not in motion`. Given that there are ten objects in the scene, this task is much more complex than the surgical case.

To demonstrate the value of this approach, we use our virtual sensor data to improve temporal segmentation and

*equal contribution
[1]Technische Universität München, Germany
[2]Johns Hopkins University, Baltimore, USA
[3]University of Bologna, Italy

With the support of the Technische Universität München - Institute for Advanced Study, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant no 291763. Also with support from National Science Foundation grant NRI-1227277

[1]`http://www.conceptkitchen2025.com/`

classification of fine-grained actions. More specifically, we extend the model of Lea et al. [5], which solely uses sensor data, to enable video-based recognition on these tasks. They introduce a type of action primitive that captures how sensors change throughout the course of an action. The model, a Latent Convolutional Skip Chain CRF (LC-SC-CRF) captures actions, transitions between actions, and temporal priors. We modify their sensor-based action primitives to incorporate video data by means of a learning approach based on a Convolutional Neural Network (CNN).

On JIGSAWS our recognition results are as good as the kinematics we learn on and are superior to the state of the art video-based results. In addition, we perform experiments to measure how well these models transfer from one task to another. For example, we train a CNN using data from a Suturing task and evaluate on a Needle Passing task.

Finally, we employ two recent CNN visualization techniques to highlight what our model learns. We look at correlations between the internal activations and different actions as well as correlations between certain (predicted) sensor signals and the image content.

To summarize, our contributions are:

1) We develop a CNN architecture that predicts continuous and discrete sensor signals from video frames for use in an action recognition model.
2) We show that predicting sensor-data improves video-based action recognition performance on two datasets. Furthermore, we show that CNN models trained on one task generalize to other tasks.
3) We highlight the capabilities of the CNN using two visualization techniques.

II. Background and Related Work

First we motivate the importance of our application domains and then describe recent work on video analysis.

Surgical Analysis: In recent years, hospitals have collected a deluge of data from surgical training and live procedures. There have been several calls for improving surgical education through quantitative analysis [6], [7]. New methods for automated analysis of both live surgery and training tasks could improve both automatic surgical workflow understanding and educational practice through computer-assisted training.

While there has been much work on automated assessment, a large body of it has been devoted to robotic surgery using systems like the da Vinci robot. While useful for robotic applications, it ignores the large swaths of data comprised solely of video sequences like from traditional laparoscopy. One goal of this work is to extend these robotic methods to non-robotic settings.

Previous work on activity recognition for surgical robot training evaluation has primarily relied on using the kinematics of the surgical tools to segment and classify each step in a training exercise. Early work on the JIGSAWS dataset by Lin et al. [8] extracts local time series signals and applies Linear Discriminant Analysis to classify motions. Varadarajan [9] used variations on Hidden Markov Models (HMMs) and Linear Dynamical Systems (LDSs) in his thesis to model a high dimensional feature vector including tool positions, velocities, joint rotations, gripper states, and other robot configurations states. Complementary research uses the video data from each trial [10], [1]. In [1] Tao et al. use a bag-of-features approach with video using spatio-temporal interest points (STIPs) and achieves worse accuracy than kinematics-based approaches. Lea et al. [2] learns a spatiotemporal CNN to recognize surgical actions solely with video. Other related works include daVinci tool tracking [11] and skills assessment [12], [13].

Cooking tasks: There have been several recent works focused on the cooking domain [14], [15], [16]. Here, we use the 50 Salads dataset of Stein and McKenna [16]. Vision-based work for cooking tasks has focused on leveraging human pose. For example [17] and [18] use the predicted body configuration to predict a sequence of actions. Other object-centric methods (e.g. [19], [20]) localize objects in each image and use their identities and relationships to predict actions. A downside to object-centric approaches is that they require knowing the specific set of objects being used. For example, if the system was only trained on green apples then it would not be able to correctly recognize red apples. In contrast, our sensor-based methods are applicable regardless of which specific object is in use because we focus on generic tools.

Video Analysis: Deep learning, specifically using CNNs, has become an important machine learning tool in recent years. In the computer vision literature, deep learning has been applied to many regression tasks. In human pose estimation (e.g. [21], [22], [23]), CNNs are used to regress 2D or 3D body joint positions given an image crop of a human. In general object detection, recent methods like Faster R-CNN [24] and YOLO [25] use regression CNNs to detect multiple objects in an image. For facial landmark detection (e.g. [26]) predict the location of salient keypoints on a given face using CNNs. In text recognition, Jaderberg et al. [27] developed a regression-based CNN to refine a precise bounding box prediction for extracting words in images. While our task is very similar to these, nobody has exploited sensor data, as we do in this paper, to learn improved CNN representations.

There have been numerous papers from both the computer vision and robotics communities modeling actions. In computer vision, most papers focus on action classification. Many of these methods rely on bag-of-words models using spatio-temporal features like Improved Dense Trajectories (IDT) [28]. CNN-based methods are becoming more common but are often used in conjunction with IDTs [29]. Zisserman et al. [30] developed a CNN architecture using color and optical flow learns patterns similar to IDT. Additional recent work combines CNNs with Recurrent Neural Networks (RNNs) for video analysis. Wu et al. [31] propose an RNN with Long Short Term Memory (LSTM) for action classification using video and optical flow.

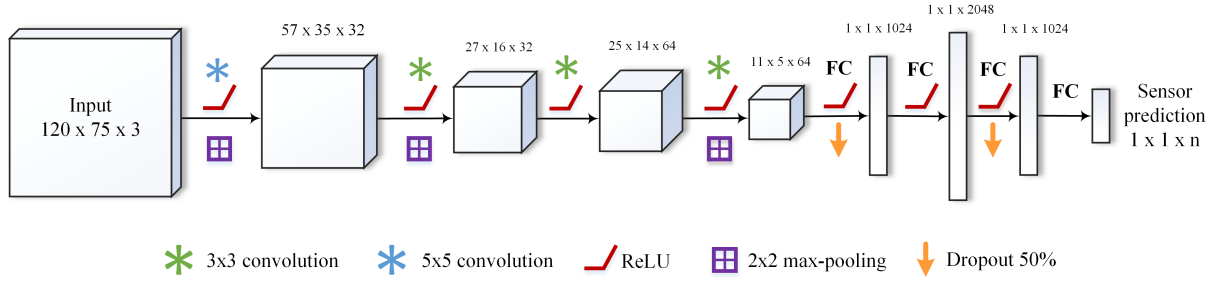

Fig. 2: The architecture of the CNN used to learn the mapping f from input image $\mathcal{I}$ to sensor data $\mathcal{S}$. The output size depends on the number n of sensor values to predict. In training, we minimize the standard L2 loss.

III. Model

Our approach is composed of two components. The first employs a CNN to predict sensor values from images. The second uses the video-predicted sensor values in tandem an extended version of the recent Latent Convolutional Skip Chain Conditional Random Field (LC-SC-CRF) [5] for action recognition.

First we define the mathematical framework which is composed of video-based action-primitives, pairwise action transitions, and temporal priors. Then we introduce the CNN architecture for learning sensor signals from video and further detail our model. Finally, we introduce the CNN feature visualization approach that we use to determine which visual activations drive the learning procedure in our network. The CNN image model is shown in Figure 2 and the temporal model is shown in Figure 3.

A. Notation and Framework

Let $\mathcal{S}_t$ be a vector of sensor values (e.g. is each object in motion) and Y_t be an action label (e.g. cutting, peeling) at time t from 1 to T. The conditional distribution of all labels $Y = \{Y_t\}$ given the signals $\mathcal{S} = \{\mathcal{S}_t\}$ is modeled using a Latent CRF with energy

$$E(\mathcal{S}, Y) = \max_h \sum_{t=1}^{T} \phi(\mathcal{S}, Y, h, t) + \psi(Y, h, t) + \pi(Y, h, t) \tag{1}$$

This energy is comprised of latent action primitives (ϕ), pairwise action transitions (ψ), and temporal priors (π). Variable h denotes the set of latent variables for each time step. Note that during the testing phase we do not have access to the true signals $\mathcal{S}$ and thus we substitute the predicted signals $\tilde{\mathcal{S}}$.

B. Predicting Sensor Values from Video

We formulate the problem of predicting sensor values from video as a regression problem. Given image $\mathcal{I}_t$, we learn a function f such that $f : \mathcal{I} \mapsto \mathcal{S}$. Let $\mathcal{I}$ be defined over the lattice $\Omega = \{1, \ldots, h\} \times \{1, \ldots, w\}$, where w and h denote the image width and height respectively. Hence, for an RGB frame the input is represented by an array of size $h \times w \times 3$.

We learn the mapping f using a CNN architecture specialized for regressing sensor data. The architecture, as seen in Figure 2, is adapted from [21]. We use four convolutional layers which hierarchically model the contents of an image. They are followed by three fully connected layers which chapture the global relationships. We differ from [21] in that we remove the normalization layers and adapt the image and convolution sizes. Experimentally we found that this performs better for our tasks.

Note that in contrast to many object-recognition models, such as those trained on ImageNet, we use far fewer filters and latent states. Those models predict 1000 object classes whereas we predict up to 10 sensor values.

We apply a linear transformation to the ground truth signals to convert them from metric units to the unit cube: the signals $\mathcal{S}$ are translated and scaled linearly by applying diagonal scaling matrix Γ and translation $\mathcal{T}$ such that $\Gamma\mathcal{S} - \mathcal{T} \in [0, 1]^n$. Γ and $\mathcal{T}$ are computed on the training set. At test-time Γ^{-1} and $\mathcal{T}^{-1}$ are used to rescale the prediction back into the original ranges. These normalization steps make the hyper-parameters (e.g. learning rate, momentum) more robust across different tasks and datasets. Furthermore, normalization helps weigh the relative importance of each individual sensor signal.

Many other papers find that normalizing the images by subtracting the mean image from the training set, $\overline{\mathcal{I}} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{I}_i$, and scaling to unit variance improves convergence. However, we found that this normalization did not improve convergence, thus we omit this step.

Furthermore, in contrast to the findings of [21], we found that using an L2 training objective performs better than Tukey's biweight loss. That work claims the biweight loss is better because it is less susceptible to outliers in the hand-labeled training data. In our case, we regress real sensor measurements which are much more accurate. In fact, we found that including all samples is important for predicting extremal sensor values.

C. Latent Action Primitives

We use the latent convolutional action primitives from Lea et al. [5] to model how sensor signals change over time. Complex actions, like moving cucumber slices from the cutting board to a salad bowl, require modeling the state changes involved in picking up the cutting board, moving it, and scraping the slices into the bowl. These action primitives are modeled such that they model how the sensors change throughout an action.

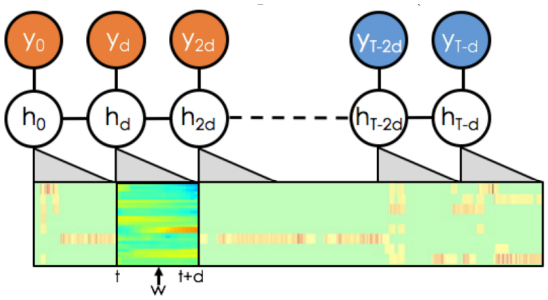

Fig. 3: We use a Latent Convolutional Skip Chain Conditional Random Field to recognize a sequence of actions from video. The bottom shows an example sequence of motion data from 50 Salads where green is off and red is high motion. The window overlaid depicts an example action primitive.

We define a set of H action primitives per action, where H is the number of latent states per class y. Each primitive, w_y^h is of size $F \times d$, where F is the number of sensors and d is the length of a primitive. These primitives are convolved with the sensor data $\mathcal{S}$ from time $t-d$ to t to get score

$$\phi(\mathcal{S}, Y, h, t) = w_{Y_t}^h \star \mathcal{S}_{t:t+d}. \tag{2}$$

Recall from the energy model that we maximize over all latent states, thus, ϕ will be the score corresponding to the best action primitive.

D. Action Transitions

The pairwise term models transitions between latent action primitives. Each action primitive is of length d, so we model the transition from times t to $t-d$. This term is a generalization of the Markov pairwise term commonly used in Hidden Markov Models and Linear Chain CRFs. This skip-term has shown empirically to perform substantially better than traditional Markov terms [32]. This term is modeled as

$$\psi(Y, h, t) = \alpha_{h_{t-d}, h_t}, \tag{3}$$

where h_t corresponds to the latent state at time t.

E. Temporal Priors

We model the likelihood of each class for each time step using a temporal prior. This is a generalization of the temporal prior used in the original LC-SC-CRF.

We define weight matrix β of size $C \times T'$, where C is the number of action classes and T' is the canonical number of time steps. Thus, each element measures the likelihood of each class occurring at each time. This prior is given by

$$\pi_s(Y, h, t) = \beta_{Y_t, t'}, \tag{4}$$

where t' is relative time $t' = T' * \frac{t}{T}$ and T is the number of timesteps in that instance. Given the limited amount of training data we set $T' = 30$ to prevent overfitting.

F. Learning

We learn parameters w, α, and β using a Structural Support Vector Machine [33]. This function minimizes the upper bound on the empirical risk. We define the SSVM loss function Δ as the Hamming distance between ground truth labeling Y^* and an arbitrary labeling Y:

$$\Delta(Y^*, Y) = \sum_{t=1}^{T} \delta(Y^{*(t)} \neq Y^{(t)}) \tag{5}$$

where δ is the Dirac delta function. We minimize this objective by applying Stochastic Gradient Descent with Adagrad [34] step updates.

For practical reasons we train the CNN independently of the LC-SC-CRF, however, in principle they could be trained jointly using the SSVM. The LC-SC-CRF is trained using predictions $\tilde{\mathcal{S}}$ as input.

Inference for our model is performed using the variation of the Viterbi decoding algorithm for skip-chains. For more details see [5] or [32].

G. Visualization

For qualitative evaluation we visualize our CNN using recent techniques proposed by Bolei et al. [35] and Jost et al. [36]. The first is applied to the 3D positions from JIGSAWS and the latter is applied to tool usage in 50 Salads.

Method 1: For continuous data we adapted the method by Bolei et al. [35] and Zeiler et al. [37] to understand which part of the image is important for the a given prediction. The method uses the following concept: given an input image $\mathcal{I}$ and its prediction $\tilde{\mathcal{S}}$ we generate many variants $\mathcal{I}_k^*$ of $\mathcal{I}$ where systematically a small patch of the image is occluded. Then for all $\mathcal{I}_k^*$ we measure the change in prediction

$$C_k = ||f(\mathcal{I}_k^*) - f(\mathcal{I})|| = ||\tilde{\mathcal{S}}_k^* - \tilde{\mathcal{S}}||. \tag{6}$$

where C_k measures the importance of the occluded patch. If the occlusion creates a large change in prediction the network deemed the original information at this location as significant. C_k can be arranged in a heatmap based on the location of the occluder.

Bolei et al. generated random occluder patches to differentiate the dominant object from background. In our tasks the camera is static, so we compute background patches deterministically based on the median image from that video. Thus, for each location, we overlay a patch of the median pixels in that region. Replacing the patch with background produces a better indicator for its importance than replacing it with random noise.

Method 2: For discrete sensors we adapt the method of Jost et al. [36]. Given an input image $\mathcal{I}$ the activation for the most confident tool is backpropagated through the network. For visual clarity, the backpropagated values through the ReLU units are only non-zero if the activation and the gradient are positive. We threshold the resulting

train	test	left x	left y	left z	right x	right y	right z
SU	**SU**	**0.43**	**0.33**	**0.54**	**0.85**	**0.88**	**0.77**
KT	SU	0.62	0.78	0.86	1.37	1.12	1.22
NP	SU	1.00	1.28	1.30	2.64	2.13	1.76
SU	KT	0.74	0.74	1.34	1.13	1.00	1.20
KT	**KT**	**0.44**	**0.39**	**0.59**	**0.52**	**0.34**	**0.67**
NP	KT	0.87	1.25	1.22	1.21	1.21	0.94
SU	NP	1.98	0.95	1.22	2.19	0.87	1.22
KT	NP	2.06	1.18	1.28	2.04	0.77	2.12
NP	**NP**	**0.80**	**0.50**	**0.95**	**0.90**	**0.65**	**0.92**

TABLE I: **Kinematics predictions RMSE in cm** Inter-task are obtained by 8-fold cross validation over the set of users and repeated for the three tasks individually.

heatmap at 50% of the maximum value to obtain focused points of attention.

H. Practical Details

We use input images with width=120 px and height=75 px using the RGB color space to train the CNN. The output is fitted into the $[0, 1]^n$ interval for training and translated and scaled back to the original range after the predictions are obtained during testing. We train the network with a batch size of 64 samples for 30 epochs using a learning rate of 0.01, a momentum of 0.9 and dropout 0.5. Initialization of all layers is done by randomly sampling from a zero-mean Gaussian distribution with variance of 10^{-2}. The CNN training is carried out using an i7 CPU with 64GB RAM and a single GEFORCE GTX 980 graphics card using the MatConvNet [38] library. Training a single fold of cross validation on JIGSAWS takes about one hour which is relatively quick compared to other deep learning methods. Each fold for the LC-SC-CRF takes around a minute.

IV. Evaluation

In this section we evaluate the benefits of augmenting videos from two substantially different datasets with virtual sensor information. We show the accuracy of the predicted sensor values and the performance in action recognition using such virtual sensor data.

A. JIGSAWS Dataset

The JIGSAWS [3] dataset consists of three tasks, Suturing (SU), Needle Passing (NP) and Knot Tying (KT), which are typically used for skills assessment on the da Vinci surgical system. For each task, eight users performed five trials resulting in about 40 sequences. The videos are accompanied with kinematics recorded from the robot in the camera coordinate frame. Our goal is to predict the 3D coordinates of the left and right gripper resulting in a six dimensional output for our network. Although the annotations in the dataset also supply the velocity of each gripper we found that predicting velocities from a single frame of video performs poorly since no temporal information is passed to the network. For applications that require velocity a simple temporal finite difference can be performed on the predicted positions.

Each task consists of approximately 10 action classes such as `reach for needle`, `insert needle`, and `loosen suture`. Each trial contains around 30 action instances and varies depending on the user and skill level.

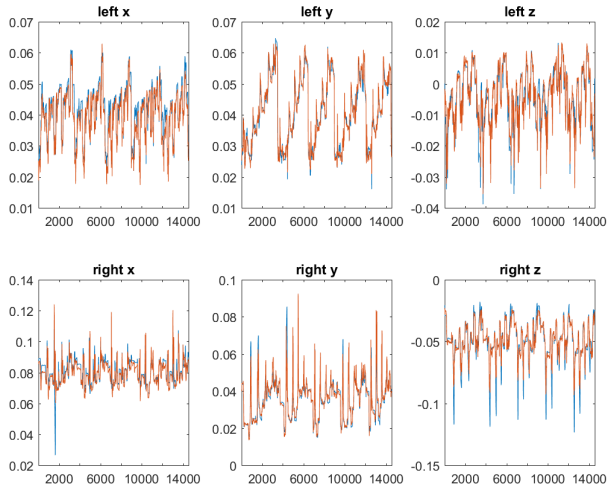

Fig. 4: The six plots show ground truth (blue) and prediction (orange) of x, y and z position of the right and left gripper (in meters). The network was trained and tested on the Suturing task.

All experiments are done using 8-fold cross validation where the model is trained on the sequences of 7 users and tested on the sequences of the left out user. All reported results are averaged over all eight folds.

For each JIGSAWS experiment we provide two experimental setups. First, we train and test our CNN using the appropriate task (e.g. Suturing, Needle Passing, or Knot Tying). Second, we train on each one of the tasks and test on the other two tasks. For example, we train on suturing and test on knot tying. These results show the ability of our model to generalize across tasks.

Kinematics Prediction: Table I shows the RMS error in centimeters for all combinations of training and test sets. Figure 4 shows the predicted positions and ground truth plots of the x, y and z coordinates of each gripper for a Suturing trial. The prediction typically follows the ground truth closely except when it underestimates spikes in position.

When performing cross-task experiments we observe a transformation in the coordinate system that the CNN cannot predict. Within each task the relative location of the grippers and objects is relatively constant. For example, for suturing the grippers typically stay near the insertion points. However, between tasks the relative geometry tends to change. Qualitatively, Figure 5 shows that there is a rigid offset and scaling between the predicted locations and ground truth. The demonstrated example is Knot Tying but similar observations have been made in other cross-task experiments as well. Note that the relative motion is predicted accurately and globally consistent, which is much more important for action recognition than the absolute values.

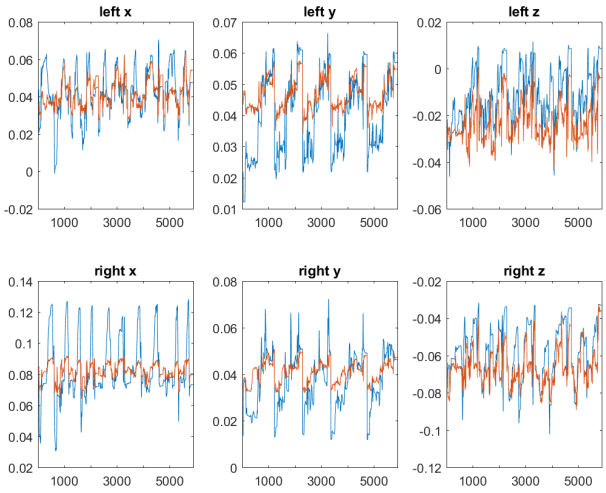

Fig. 5: Predictions (orange) of a network trained on suturing tested on a sequence of the knot tying task. Ground truth (in meters) is shown in blue. We see that there is a shift in the global coordinate system between the two tasks but still the relative motion is predicted accurately.

Test Task	GT	SU	KT	NP	[39]
Suturing	76.1	**76.6**	65.8	61.2	71.7
Knot Tying	74.2	70.3	**76.3**	67.8	66.9
Needle Passing	62.3	51.3	54.2	**60.4**	60.3

TABLE II: Action recognition results on JIGSAWS. Columns refer to models trained on ground truth kinematics (GT) or predicted from a CNN trained on each task (SU, KT, or NP).

Action Recognition: For activity recognition, we use the predicted gripper positions as input into our LC-SC-CRF. We define accuracy as the frame-wise accuracy between the ground truth and predicted labels.

Table II shows action recognition results of our model trained and tested on all nine combination of the three tasks. Results using the ground truth (GT) kinematics instead of the predicted ones as input into the LC-SC-CRF are also reported. For a video-based comparison, we show the results of Tao et al. [39], who follow a Bag of Words (BoW) approach with space-time interest points (STIPs) and use a Markov Semi-Markov Model. Our results are superior to this approach and only requires sensor data to train the CNN.

On all three tasks we outperform [39]. On Suturing and Knot Tying we perform notably better and for Needle Passing we achieve comparable results. Needle Passing is visually very different from the other two tasks which makes it more challenging to learn the appropriate information. This also explains why training on Needle Passing gives slightly higher error in Table I than when training on the other tasks.

Furthermore, we observe an interesting fact. For SU and KT we achieve a higher accuracy using the predicted kinematics than using the ground truth signals from the robot. To explain this, we observe that the inter-task shifts in the kinematics coordinate system also manifest (less significantly) in between the different users within a task. We believe that, since the CNN cannot learn this transformation, the prediction of the kinematics from the CNN averages the biases between users and thus creates a more coherent set of sensor values to train and test on.

CNN Visualization: Here we show the visualization adapted from Bolei et al. [35]. Figure 6 depicts the heatmap for the input image (middle) once by computing C_k of the subset of $\mathcal{S}$ that corresponds to the left gripper's 3D location (left heatmap) and once for the right gripper's (right heatmap). The left image clearly highlights the left gripper, implying that the position is a function of the image content in that region. Interestingly, the right image highlights both grippers, implying that the right position is a function of both gripper regions.

B. 50 Salads Dataset

The 50 Salads dataset [4] consists of 25 people each preparing salads in two trials. The scene is observed by an RGBD camera of which we use the color images. The ten tools used to prepare the meals are equipped with accelerometers employed to measure forces that move the tools. The tools are `pepper dispenser`, `bowl`, `oil bottle`, `large spoon`, `dressing glass`, `knife`, `peeler`, `small spoon`, `plate` and `chopping board`. It is unreasonable to except that the raw acceleration could be learned from video due to issues like rotational symmetry. The same movement can manifest itself in different ways visually depending on how the tool is rotated. Thus, we threshold the temporal gradient magnitude to detect whether a tool is currently in use.

50 Salads consists of several action granularities. We evaluate on the "eval" granularity which is composed of ten actions that can reasonably be recognized from tool usage: `add dressing`, `add oil`, `add pepper`, `cut`, `mix dressing`, `mix ingredients`, `peel`, `place`, `serve salad onto plate`, and `background`. Each trial is five to ten minutes in length and is composed of around 25 action instances.

Action Recognition: The action recognition results using the predicted tool usage are summarized in Table III. Predicting tool usage for ten objects is clearly much more difficult than predicting the location of two objects in JIGSAWS. Furthermore, the network must learn additional cues such as the relative relationships between the tool and objects that interact with that tool. For example, the model should correlate hand and tool locations.

Despite the increased complexity, we achieve an accuracy of over 52% in action recognition using the tool usage predictions. For comparison we evaluated using the ground truth sensors as input into the LC-SC-CRF. We also evaluated using two recent video-based approachs. The first is comparable to the approach in Rohrbach et al. [40], which uses Improved Dense Trajectories (IDT). This Bag of Words method uses a sliding window of 30 frames with 2000 clusters on the HOG channel of IDT. This method achieves 54.28% accuracy on 50 Salads. We also extract FC6 features

Fig. 6: Heatmaps for occluding the input image with a sliding grey patch (10×10 pixels) and measuring the change in the prediction of left and right gripper.

	GT	Ours	VGG	IDT
50 Salads	73.43%	52.18%	38.30%	54.28%

TABLE III: Action recognition results on 50 Salads. *GT* results are using the ground truth tools, *Ours* uses the predicted tools, *VGG* uses a pre-trained network, and IDT refers to the Improved Dense Trajectory results.

Fig. 7: CNN visualization on an image from 50 Salads where the "large spoon" is in motion. The points of attention, highlighted with red circles, are on the tool and human's hand. The network learned a correlation between the object and the hand placement for this utensil.

from a VGG network pre-trained on ImageNet. Results are only 38.30%.

Note that the results we report using the ground truth signals are lower than the ones in [5]. This is due to the fact that we train our model using different splits.

CNN Visualization: Figure 7 illustrates points of attention derived by the visualization method based on Zeiler et al. [37] on an example 50 Salads test image. The network correctly predicts that the "large spoon" is in use. This indicates that the network has learned to detect objects and hands and uses their spatial arrangement for prediction.

C. Discussion

In the experiments we show that action recognition performance can improve when the images are annotated with virtual sensor readings learned from real data.

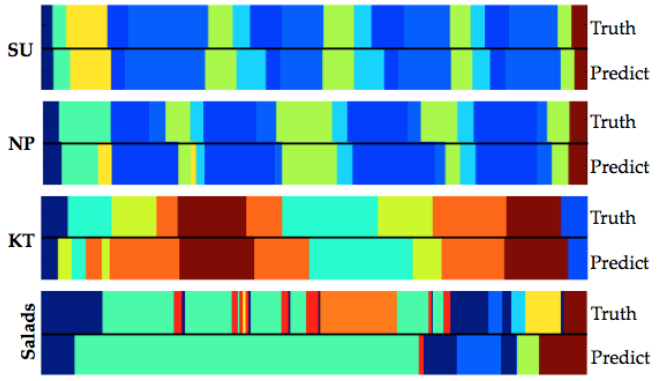

Fig. 8: Example ground truth and predicted action sequences. Each color corresponds to a different action class. The top of each plot is the set of ground truth actions over time and the bottom is predicted actions using virtual sensor data.

On JIGSAWS, when the tool positions are predicted for the same task that they are trained on, the RMS error for all tasks is sub-centimeter. Even when the network is trained on other tasks the results are all within one to two centimeters. This suggests that our model generalizes well to previously unseen tasks.

Figure 8 depicts example action predictions for each dataset and task where the colors refer to different actions. Minor errors on the JIGSAWS dataset come from temporal shifts in the prediction. On 50 Salads our model picks up on longer actions well and infers the global activity but does not capture the nuances of other, typically shorter, actions.

Naturally, our model can only regress sensors signals that can be observed or inferred from RGB images. Sensor information that is not visible in the image cannot be learned. In fact, the biggest area of improvement could come from incorporating motion information into the CNN input. For example, optical flow or a foreground model could help differentiate which parts of the image contain motion. Since tool usage is strongly coupled to motion, the performance on 50 Salads is lower relative to JIGSAWS.

V. Conclusions

We have proposed a method for sensory substitution where we regress domain-specific sensor signals from RGB video. We achieve state of the art action recognition accuracy on

JIGSAWS and show that predicted kinematics can be even better proxies for action recognition than the real measurements. Moreover, we show that our model generalizes to unseen tasks in the same domain. When challenged with a much more complex environment, like 50 Salads, we achieved similar action recognition accuracies as the state of the art using video only. Finally, we validated that the CNN is capable of learning a meaningful representation from the data in both domains by visualizing the network with two different methods.

REFERENCES

[1] L. Tao, L. Zappella, G. D. Hager, and R. Vidal, "Surgical gesture segmentation and recognition." in *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2013.

[2] C. Lea, A. Reiter, R. Vidal, and G. D. Hager, "Segmental spatiotemporal cnns for fine-grained action segmentation," *CoRR*, 2016. [Online]. Available: http://arxiv.org/abs/1602.02995

[3] Y. Gao, S. Vedula, C. Reiley, N. Ahmidi, B. Varadarajan, H. Lin, L. Tao, L. Zappella, B. Bejar, D. Yuh, *et al.*, "The jhu-isi gesture and skill assessment dataset (jigsaws): A surgical activity working set for human motion modeling," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI) M2CAI Workshop*, 2014.

[4] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2013), Zurich, Switzerland*. ACM, September 2013.

[5] C. Lea, R. Vidal, and G. Hager, "Learning convolutional action primitives for fine-grained action recognition," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[6] C. B. Barden, M. C. Specht, M. D. McCarter, J. M. Daly, and T. J. Fahey, "Effects of limited work hours on surgical training," *J. Am. Coll. Surg.*, no. 4, Oct 2002.

[7] K. Cleary, A. Kinsella, and S. K. Mun, "OR 2020 workshop report: Operating room of the future," 2005, CARS: Computer Assisted Radiology and Surgery.

[8] H. C. Lin, I. Shafran, D. Yuh, and G. D. Hager, "Towards automatic skill evaluation: detection and segmentation of robot-assisted surgical motions." *Computer aided surgery : official journal of the International Society for Computer Aided Surgery*, no. 5, sep 2006.

[9] B. Varadarajan, "Learning and inference algorithms for dynamical system models of dextrous motion," Ph.D. dissertation, Johns Hopkins University, 2011.

[10] B. B. Haro, L. Zappella, and R. Vidal, "Surgical gesture classification from video data." *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2012.

[11] S. Kumar, M. S. Narayanan, P. Singhal, J. J. Corso, and V. Krovi, "Product of tracking experts for visual tracking of surgical tools," *IEEE International Conference on Automation Science and Engineering (CASE)*, 2013.

[12] S.-K. Jun, M. S. Narayanan, P. Agarwal, A. Eddib, P. Singhal, S. Garimella, and V. Krovi, "Robotic Minimally Invasive Surgical skill assessment based on automated video-analysis motion studies," *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, jun 2012.

[13] A. Zia, Y. Sharma, V. Bettadapura, E. L. Sarin, M. A. Clements, and I. Essa, "Automated assessment of surgical skills using frequency analysis," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015*, 2015.

[14] K. Ramirez-Amaro, M. Beetz, and G. Cheng, "Automatic segmentation and recognition of human activities from observation based on semantic reasoning," in *IEEE Conference on Intelligent Robots and Systems (IROS)*, 2014.

[15] J. Lei, X. Ren, and D. Fox, "Fine-grained kitchen activity recognition using RGB-D," in *ACM Conference on Ubiquitous Computing (Ubicomp)*, 2012.

[16] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. ACM, September 2013.

[17] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele, "A database for fine grained activity detection of cooking activities," in *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, 2012, pp. 1194–1201.

[18] G. Cheron, I. Laptev, and C. Schmid, "P-cnn: Pose-based cnn features for action recognition," 2015.

[19] B. Ni, V. R. Paramathayalan, and P. Moulin, "Multiple granularity analysis for fine-grained action detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[20] A. Fathi and J. M. Rehg, "Modeling actions through state changes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[21] V. Belagiannis, C. Rupprecht, G. Carneiro, and N. Navab, "Robust optimization for deep regression," in *International Conference on Computer Vision (ICCV)*. IEEE, December 2015.

[22] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2014, pp. 1653–1660.

[23] T. Pfister, K. Simonyan, J. Charles, and A. Zisserman, "Deep convolutional neural networks for efficient pose estimation in gesture videos," in *Computer Vision–ACCV 2014*. Springer, 2015, pp. 538–552.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Neural Information Processing Systems (NIPS)*, 2015.

[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[26] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *IEEE European Conference on Computer Vision (ECCV)*, 2014.

[27] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *International Journal of Computer Vision*, pp. 1–20, 2014.

[28] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *IEEE International Conference on Computer Vision (ICCV)*, Sydney, Australia, 2013.

[29] M. Jain, J. C. van Gemert, and C. G. M. Snoek, "What do 15,000 object categories tell us about classifying and localizing actions?" in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[30] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," pp. 568–576, 2014.

[31] Z. Wu, X. Wang, Y. Jiang, H. Ye, and X. Xue, "Modeling spatial-temporal clues in a hybrid deep learning framework for video classification," in *ACM Conference on Multimedia*, 2015, pp. 461–470.

[32] C. Lea, G. D. Hager, and R. Vidal, "An improved model for segmentation and recognition of fine-grained activities with application to surgical training tasks," in *2015 IEEE Winter Conference on Applications of Computer Vision, WACV*, 2015, pp. 1123–1129.

[33] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large Margin Methods for Structured and Interdependent Output Variables," *jmlr*, vol. 6, pp. 1453–1484, 2005.

[34] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," EECS Department, University of California, Berkeley, Tech. Rep., 2010.

[35] Z. Bolei, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," 2015.

[36] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller, "Striving for simplicity: The all convolutional net," *CoRR*, vol. abs/1412.6806, 2014.

[37] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer vision–ECCV 2014*, 2014.

[38] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," *CoRR*, vol. abs/1412.4564, 2014.

[39] L. Tao, L. Zappella, G. D. Hager, and R. Vidal, "Surgical gesture segmentation and recognition," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*. Springer, 2013.

[40] M. Rohrbach, A. Rohrbach, M. Regneri, S. Amin, M. Andriluka, M. Pinkal, and B. Schiele, "Recognizing fine-grained and composite activities using hand-centric features and script data," *International Journal of Computer Vision (IJCV)*, 2015.