# TACTILE GRAPHS FOR GRASP STABILITY PREDICTION

**Brayan S. Zapata-Impata*** **& Pablo Gil**
Automatics, Robotics, and Artificial Vision Lab (AUROVA)
University of Alicante
Alicante, Spain
{brayan.impata, pablo.gil}@ua.es

**Alberto Garcia-Garcia*, Sergio Orts-Escolano & Jose Garcia-Rodriguez**
3D Perception Lab (3DPL)
University of Alicante
Alicante, Spain
{agarcia, jgarcia}@dtic.ua.es, sorts@ua.es

## ABSTRACT

In robotics, tactile sensors provide useful contact data during the interaction with an object which can be used to determine the stability of a grasp. Previous works represented tactile readings as plain feature vectors or matrix-like tactile images. In this work, we explore an alternative way of exploiting tactile information to predict grasp stability by leveraging graph-like representations of tactile data, which preserve the actual spatial arrangement of the sensor's taxels and their locality.

## 1 INTRODUCTION

When we humans grasp objects, we can accurately predict the stability of the grasp using our tactile sense, along with our vision and other senses. This skill is desirable for any robotic manipulator since it favors the early detection of failures. In order to approach a solution to the problem of predicting grasp stability, tactile sensors are being used as the main source of data since they provide valuable information about the acting forces during the interaction of the hand with the objects (Kappassov et al., 2015). As for the stability prediction, two states are usually distinguished: stable, meaning that the object is firmly grasped; or slippery, meaning that the object could slide from the hand.

In the last years, deep learning models are being applied to the problem of grasp stability prediction using tactile sensors as input. Kwiatkowski et al. (2017) built a composite image by placing the readings of two matrix-like sensors side by side. Then, they used this tactile image as input for a Convolutional Neural Network (CNN) along with the proprioceptive data from the robot. As a result, the proposed method calculated by itself the features needed for predicting grasp stability. In some other cases, the tactile sensor is not naturally arranged in an array so a pre-processing is necessary. For example, Zapata-Impata et al. (2018) studied how the readings from a non-matrix like sensor should be arranged in a matrix in order to train a CNN for grasp stability prediction. Although their work showed promising results, the spatial distribution of the sensor was not accurately reflected because it reduced the 3D locations of the taxels into 2D coordinates of an image.

Lately, Graph Neural Networks (GNNs) have emerged as an alternative to process irregular data which can be structured as graphs. Various works have successfully made use of them to deal with unstructured 3D representations mainly in classification tasks. Most of them have proposed extensions to the well-known CNN architecture to process graph-structured data (Ying et al., 2018), (Zhang et al., 2018) and others have also extended Recurrent Neural Network (RNN) architectures to support inference on graph sequences (Li et al., 2015; Sanchez-Gonzalez et al., 2018).

However, the generalization of those traditional architectures is not trivial since various problems must be addressed when applying convolution filters in domains in which there is no regular structure. In that regard, there are two dominant ways to convolve a graph signal with a learned filter:

*Authors contributed equally.

spatial or spectral. On the one hand, spectral methods are characterized by providing a spectral graph theoretical formulation of CNN on graphs using Graph Signal Processing (GSP) theory (Shuman et al., 2013). On the other hand, spatial methods (Bruna et al., 2013; Kipf & Welling, 2016; Defferrard et al., 2016) constitute the straightforward generalization of convolutions to graphs, just by sliding a filter on the vertices as a traditional CNN does with any other structured data representation. Nevertheless, the borders between those two approaches are being progressively blurred thanks to the unification provided by message passing networks (Gilmer et al., 2017).

Following the success of GNNs in various fields in general and in certain robotic tasks in particular, such as robot team coordination (Prorok, 2018) and policy learning (Wang et al., 2018), we intend to use a GNN to process tactile sensor readings and predict grasp stability. By doing so, we expect that such architecture is able to better capture the spatial locality and relationships of the tactile sensor readings expressed as graphs. In contrast with previous works, in which tactile readings are processed as classic signals, we propose the use of graphs for representing tactile data. Graphs can reflect more accurately the real distribution of the electrodes in the sensor as well as their spatial relationships, which should be of great value for learning tactile features. Our contributions are as follows:

- We process tactile readings differently: instead of considering them as 1D arrays or 2D images, we build a 3D graph connecting the sensing points of the sensor.
- We quantitatively check the performance of this new methodology in the real world using a set of tactile sensors installed in a robotic hand.
- We release an extension (with more objects and grasps configurations) of a dataset for robotic grasp stability prediction using tactile perception.

## 2 Proposal

### 2.1 Robotic Set Up

In this work, we use the BioTac SP tactile sensors developed by Syntouch (2018). The sensor provides three different sensory modalities: force, pressure and temperature. In more detail, this biomimetic sensor counts with 24 electrodes, also named taxels, integrated in just a single phalanx. For our work, we use a setup of three BioTac SP sensors in the tip of the index, middle finger and thumb of a Shadow Dexterous robotic hand developed by Shadow Robot Company (2018).

### 2.2 Tactile Graphs

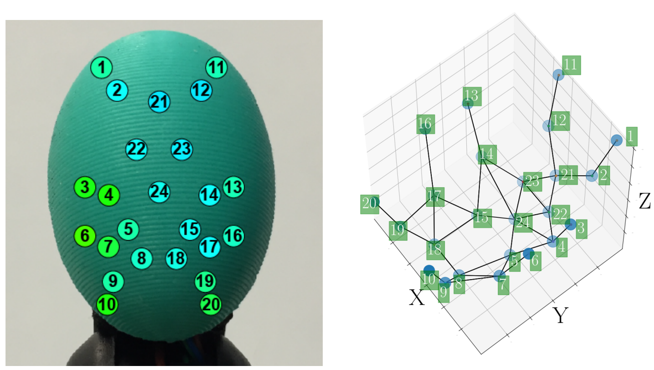


Figure 1: (left) BioTac SP tactile sensor with its 24 electrodes approximated position. (right) 3D visualization of the tactile graph layout using the accurate spatial arrangement from the actual sensor.

In order to feed our Graph Neural Network, we expressed the aforementioned sensor readings in a novel graph representation, namely tactile graphs. Such graphs are triplet $G = (N, E, Y)$ where $N$ is a set of 24 nodes $n_0, ..., n_{23}$ (one for each electrode or taxel in the sensor), $E$ is a set of ordered pair of vertices called edges, and $Y$ is the label or class of the graph (in our case, stable or unstable).

Each node $n$ in the graph $G$ is characterized by a 3D position $p_n = (x_n, y_n, z_n)$ and a feature vector $f_n = (f_{n_0}, ..., f_{n_F})$ of arbitrary length $F$. Node positions $p_n$ are accurately mapped to the physical taxel $(X, Y, Z)$ coordinates within the sensor, provided by the manufacturer. Edges or connections are generated following two different approaches: manual or k-Nearest Neighbors (k-NN). For the first approach, we manually specified undirected connections following proximity and symmetry criteria. For the second one, we generated directed edges towards each k-Nearest Neighbors for each node. Figure 1 shows a BioTac SP sensor and a 3D tactile graph representing its taxels. Node features $f_n$ correspond to the taxel pressure readings. In our case, each node has three features, i.e., the pressure reading for each finger: index $f_{n_0}$, middle $f_{n_1}$, and thumb $f_{n_2}$. Figure 2 shows visualizations of the three features for a sample generated with $k = 0$ (manually defined edges)

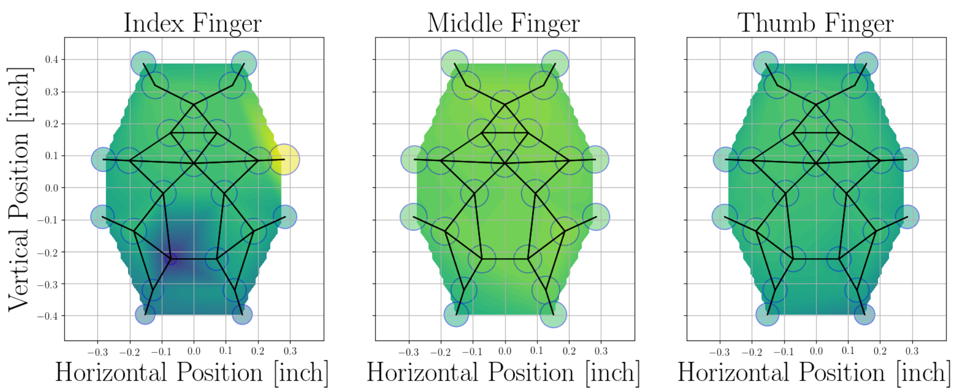


Figure 2: Un-directed tactile graphs generated with manually defined edges. The three features $f_{n_0}$, $f_{n_1}$, and $f_{n_2}$ are decoupled into three plots and represented as contour plots in the XY plane.

### 2.3 Graph Neural Network

Our Graph Neural Network (GNN) of choice is based on the Graph Convolutional Network (GCN) model by Kipf & Welling (2016). The goal of such model is to learn features on a graph $G = (N, E, Y)$ by taking as input a feature matrix $X$ ($N \times F$ with a feature vector $f_n$ for each node $n$) and a description of the graph structure in the shape of an adjacency matrix $A$. The output is another feature matrix $Z$ ($N \times F'$ with node-level feature vectors $f'_n$ with a predefined number of output features $F'$). Each GCN layer $H^{(l)}$ in a network with $L$ layers can be expressed as a non-linear function $H^{(l+1)} = f(H^{(l)}, A)$. The first layer takes the input feature matrix ($H^{(0)} = X$) and the final layer generates the output node-level feature matrix ($Z = H^{(L)}$). Each intermediate layer generates a node-level feature matrix $Z^{(l)}$ which is fed to the next layer. In the case of Kipf & Welling (2016), the graph-convolution layer $f(H^{(l)}, A)$ is defined, in the most basic instantiation, as $\sigma(AH^{(l)}W^{(l)})$, where $\sigma$ is an activation function and $W^{(l)}$ is the weight matrix for the $l$ layer.

This basic framework was heavily extended to overcome two limitations: (1) unless there are explicitly defined self-loops in the graph, the multiplication of $A$ only sums up the feature vectors of all the neighboring nodes but not the node itself, and (2) since $A$ is not normalized by default, the multiplication of $A$ has a huge impact on the scale of the feature vectors. Those two improvements form the layer propagation rule Kipf & Welling (2016): $f(H^{(l)}, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$. This is the *GCNConv* operator we used to build our GNN.

However, it is important to remark again that this model produces a feature matrix with node-level feature vectors yet our problem needs to classify the whole graph either as stable or slippery. To produce such binary graph-level classification output we introduced pooling operations to reduce the amount of nodes in the graph and fully connected layers to perform high-level reasoning. However, after preliminary experiments with *Graclus* **?** and *Voxel Grid* **?** pooling, we found out that our model performed better without them so we removed those layers for our final experiments. We hypothesize that our spatial size is already small enough to reduce it even more.

# 3 EXPERIMENTAL EVALUATION

The dataset used in our experiments contains grasp samples performed over $51$ objects with different geometries (i.e. cylinders, spheres, boxes), materials (i.e. wood, plastic, aluminum), stiffness levels (i.e. solid, soft) as well as sizes and weights. There are three hand configurations used for recording the grasp data: *palm down* grasps were performed pointing the palm of the hand downwards, *palm side* grasps were recorded pointing it to one side, with the thumb upwards, and *palm 45* which is in between the other two configurations at an angle of $45$ degrees. Table 1 provides a quantitative summary of the dataset for both splits and all configurations (see Supplementary Material for more detailed information).

Table 1: Summary of the tactile dataset used in this work to validate our architecture.

| | **Training Set** | | **Test Set** | |
|---|---|---|---|---|
| **Configuration** | **Stable** | **Slippery** | **Stable** | **Slippery** |
| Palm Down | 667 | 609 | 153 | 163 |
| Palm Side | 603 | 670 | 157 | 165 |
| Palm 45 | 1058 | 1075 | 250 | 261 |

We investigated the impact of network depth (convolution layers) and width (amount of features per layer). To that end, we have tested ten different models ranging from one to ten *GCNConv* layers with increasing number of features $(8, 16, 32, 48, 64)$. Rectified Linear Unit (ReLU) activations were used after each convolutional layer. Two fully connected layers were also placed at the end of the network (with $128$ and $2$ output features respectively) to produce the classification result. We used the manually defined graph connections $(k = 0)$. Figure 3 (left) shows the results of this set of experiments.

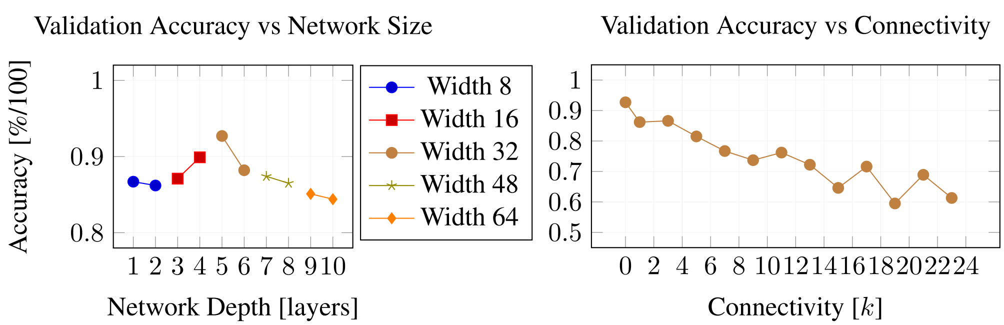


Figure 3: Performance according to network size (left) graph connectivity (right).

We experimented with manually specified edges $(k = 0)$ and the k-NN strategy with $k = [1, 23]$. As shown in Figure 3 (right), the performance of the network degraded as the connectivity of the graph increased. Using the k-NN strategy, smaller $k$ values achieved greater performance in terms of validation accuracy. However, none of them improved the performance $(92.7\%)$ yielded by the network trained with the graph created using the manual connectivity $(k = 0)$.

In order to prove the generalization capabilities of our system, we trained our best network (5 layers whose widths were $8 - 8 - 16 - 16 - 32$ and $k = 0$) with our whole training set and evaluated it on the test sets. All results are reported in Table 2. There is a significant drop in accuracy when dealing with completely unknown objects. Recall that the test set consists of new objects with different geometries and stiffness levels so they are substantially different from the training set, even though the hand orientations were the same as in the training set. We carried out the same generalization test with the best CNN configuration found by Zapata-Impata et al. (2018), using it as baseline for comparison purposes. More precisely, we trained a CNN with 32 convolutional 3x3 filters followed by a ReLU and a fully connected layer of 1024 ReLUs. Then, this layer was connected to a softmax layer in order to classify grasps. The input for this network were 2D tactile images concatenated in a 12x11x3 tensor. Results are presented in Table 3.

Table 2: Results of generalization experiments with the proposed TactileGCN.

| **Test Set** | **Accuracy** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|
| Down | 0.741 | 0.741 | 0.751 | 0.745 |
| 45 | 0.774 | 0.774 | 0.783 | 0.778 |
| Side | 0.751 | 0.785 | 0.709 | 0.745 |

Table 3: Results of generalization experiments on the testing splits with the baseline CNN.

| **Test Set** | **Accuracy** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|
| Down | 0.829 | 0.936 | 0.717 | 0.812 |
| 45 | 0.763 | 0.860 | 0.639 | 0.734 |
| Side | 0.708 | 0.858 | 0.515 | 0.643 |

Results from Table 2 and Table 3 suggest that GCNs have a more robust performance with an average F1 score between test sets of $75.6\% \pm 1.1\%$. In contrast, the CNN showed a less stable performance with lower average F1 rate ($72.9\% \pm 6.9\%$). This baseline model learned much better how to predict grasps stability from grasps in the *palm down* orientation, even achieving higher rates than the GCN. However, its performance dropped on the other two orientations. Since the input to the CNN was a generated 2D tactile image, orientations highly affected the distribution of stability patterns inside of it. Thanks to the more appropriate representation given by graphs, the GCN seems to be robust to these changes in hand's orientations, resulting in similar performance rates independently of the test set.

The implementation of our architecture is fully available at the following GitHub repository with open-source license: `https://github.com/3dperceptionlab/tactile-gcn`.

## 4 Conclusion, Limitations, and Future Works

Graph representations of tactile readings can be successfully used for learning the task of grasp stability prediction. Nevertheless, there are some problems that must be solved: (1) defining a proper graph connectivity, (2) GCNs proved to be data hungry models for learning, (3) generalization to radically new objects has still a lot of room for improvement. As a future work, we also plan to decouple the currently unified GCN for the three fingers so that each graph is processed by a different network path. Furthermore, we plan to model the noise of each individual taxel and augment each sample on the fly by adding random noise following each taxel's distribution. At last, we plan to extend the architecture to predict grasp stability over temporal sequences by fusing the GCN model with others like Long Short-Term Memory Network (LSTM).

### Acknowledgments

Experiments were made possible by generous hardware donation from NVIDIA (Titan Maxwell). This work has been funded by the Spanish Government with Feder funds (TIN2016-76515-R), by two grants for PhD studies (FPU15/04516, BES-2016-07829), by regional projects (GV/2018/022, GRE16-19) and by the European Commission (COMMANDIA SOE2/P1/F0638), action supported by Interreg-V Sudoe.

## REFERENCES

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pp. 3844–3852, 2016.

Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence*, 29(11): 1944–1957, 2007.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1263–1272. JMLR. org, 2017.

Zhanat Kappassov, Juan-Antonio Corrales, and Véronique Perdereau. Tactile sensing in dexterous robot hands — Review. *Robotics and Autonomous Systems*, 74:195–220, dec 2015. ISSN 09218890. doi: 10.1016/j.robot.2015.07.015.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Jennifer Kwiatkowski, Deen Cockburn, and Vincent Duchaine. Grasp stability assessment through the fusion of proprioception and tactile signals using convolutional neural networks. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pp. 286–292. IEEE, sep 2017. ISBN 978-1-5386-2682-5. doi: 10.1109/IROS.2017.8202170.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Amanda Prorok. Graph neural networks for learning robot team coordination. *arXiv preprint arXiv:1805.03737*, 2018.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. *arXiv preprint arXiv:1806.01242*, 2018.

Shadow Robot Company. Shadow Dexterous Hand, 2018. http://www.shadowrobot.com/products/dexterous-hand/.

David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

Martin Simonovsky and Nikos Komodakis. Dynamic edgeconditioned filters in convolutional neural networks on graphs. In *Proc. CVPR*, 2017.

Syntouch. BioTac SP, 2018. https://www.syntouchinc.com/en/sensor-technology/.

Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. Nervenet: Learning structured policy with graph neural networks. 2018.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *Advances in Neural Information Processing Systems*, pp. 4800–4810, 2018.

Brayan S. Zapata-Impata, Pablo Gil, and Fernando Torres Medina. Non-matrix tactile sensors: How can be exploited their local connectivity for predicting grasp stability? *IEEE/RSJ IROS 2018 Workshop RoboTac: New Progress in Tactile Perception and Learning in Robotics*, abs/1809.05551, 2018.

Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.