IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 September 2023, accepted 6 October 2023, date of publication 9 October 2023, date of current version 18 October 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3323220*

SURVEY

# Graph Learning in Robotics: A Survey

**FRANCESCA PISTILLI, (Member, IEEE), AND GIUSEPPE AVERTA, (Member, IEEE)**
Department of Control and Computer Engineer, Polytechnic of Turin, 10129 Turin, Italy

Corresponding author: Francesca Pistilli (francesca.pistilli@polito.it)

This work was supported in part by the Future Artificial Intelligence Research (FAIR), and in part by the European Union Next-GenerationEU [PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR)—MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3-D.D. 1555 11/10/2022] under Grant PE00000013.

**ABSTRACT** Deep neural networks for graphs have emerged as a powerful tool for learning on complex non-euclidean data, which is becoming increasingly common for a variety of different applications. Yet, although their potential has been widely recognised in the machine learning community, graph learning is largely unexplored for downstream tasks such as robotics applications. To fully unlock their potential, hence, we propose a review of graph neural architectures from a robotics perspective. The paper covers the fundamentals of graph-based models, including their architecture, training procedures, and applications. It also discusses recent advancements and challenges that arise in applied settings, related for example to the integration of perception, decision-making, and control. Finally, the paper provides an extensive review of various robotic applications that benefit from learning on graph structures, such as bodies and contacts modelling, robotic manipulation, action recognition, fleet motion planning, and many more. This survey aims to provide readers with a thorough understanding of the capabilities and limitations of graph neural architectures in robotics, and to highlight potential avenues for future research.

**INDEX TERMS** Graph neural networks, robotics, deep learning, human–machine interaction.

## I. INTRODUCTION

Over the last few years, we observed an exponential growth of deep learning methods for a variety of different data modalities and applications. Thanks to its closeness to real world applications, robotics represents the perfect ultimate harbour where deep learning methods find their downstream task.

Notable examples are the field of robot vision, and the upcoming trend of foundational multi-modal models for task and motion planning, where images are coupled with other information sources such as text and audio. However, in many practical cases, a mere representation of knowledge through bi-dimensional pixels or temporal sequence of tokens may not be sufficient to properly convey the information. Three-dimensional visual data, functional and geometrical relationships, and interaction between multiple agents and objects - for example - require more complex and unstructured data representation (see Fig. 1). One popular approach to handle these cases is through graphs encoding, where information is represented as an organised set of nodes and edges, embedding relevant features of the task.

Noteworthy, very often we observe a significant delay between when new methods are proposed in the machine learning and computer vision communities, and their application in robotics-oriented use-cases. Therefore, the purpose of this survey is to lay down the current state of the art on graph learning, with specific focus on their potential application on robotic tasks, with the aim of collecting the available knowledge at the moment, and showcase the potential of graph theory and learning, to foster future developments of intelligent machines able to learn on complex data structures.

### A. LEARNING FROM UNSTRUCTURED DATA

Graph neural networks are powerful mathematical tools for knowledge representation, able to encode structured and unstructured information in a convenient fashion, through nodes which model entities (objects, functional elements, robots), and edges that encode their spatial, temporal or functional relationship. Graphs are commonly used to model

The associate editor coordinating the review of this manuscript and approving it for publication was Shun-Feng Su.

© 2023 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

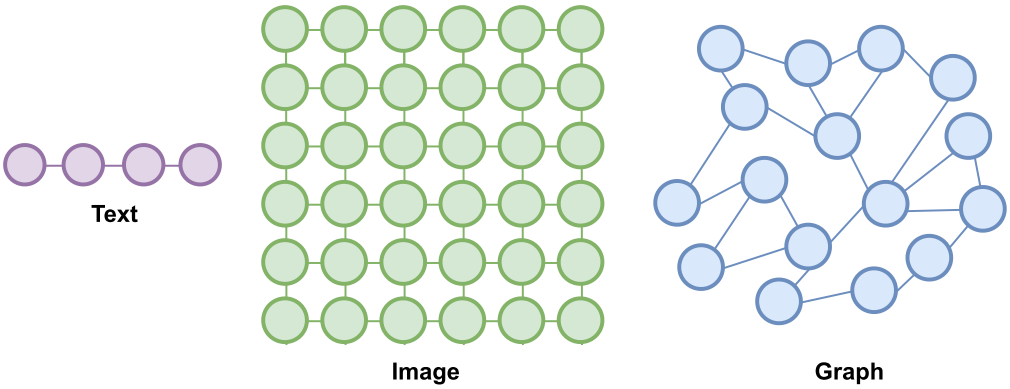



**FIGURE 1.** Schematics of the information representation encoded in the form of images, text and graphs. Although being by far the most common data structures to train artificial neural architectures, images and ordered series of tokens cannot handle unstructured data distribution. To address this challenge, graph structures constitute the primary mathematical representation for an efficient and effective data representation.

a variety of different elements, such as molecules [1] for drug discovery, physics simulations [2], scientific citations [3] or social networks [4]. In computer vision, this architecture has emerged as a promising tool to deal with data that lie on irregular domains, such as point clouds [5], [6], to uncover non-local similarities in the data [7], and video understanding [8], [9].

Thanks to these promising results, Graph Neural Networks (GNNs) have gained the attention of the robotic community, being proposed as key technological solutions for more and more new tasks and applications. As noticeable examples, it is worth mentioning grasping and manipulation, in which robots are usually tasked to process as input three-dimensional data, for which graphs can be used to provide an efficient yet effective modelling tool. Another relevant scenario is the use of graph architectures to capture and model spatial and functional keypoints of environments or workspaces, allowing robots to better understand the environment wherein, how to interact safely with objects and zones, and how to plan meaningful tasks.

GNNs also lend themselves well to reinforcement learning, as they can learn to predict future states based on current state and action signals. In robotics, this brings the benefit that GNNs can effectively plan robot actions sequences taking into account current and future states to select the best succession of action. In addition, GNNs can help robots learn how to move and behave in more complex environments by providing more detailed models of their surroundings. Deep Reinforcement Learning (DRL) has been a resourceful tool for improving decision-making tasks and control problems in robot automation [10], [11], [12]. However, in many cases DRL suffers of limited generalisation capability, with limits a seamlessly adaptation to new scenarios, different than the ones experienced at training time [13]. The ability to generalise to novel scenarios it is of paramount importance for many practical applications, and to shorten the distance between sim-to-real simulations. To address this limitation, several projects demonstrated significant benefits in the use of GNNs for learning robust policies across scenarios [10], [11], [12], benefiting from the intrinsic generalisation ability of graphs encoding.

### B. SURVEY METHODOLOGY

To ensure an high quality selection of papers for our survey, we first employed Elsevier Scopus[1] as search engine, using as keywords deep learning, robotics, graph learning and graph neural networks. This search yielded a first list of documents, from which we then selected only the manuscript published in journals with ranking Q2 or above on ScimagoJR[2] for Computer Science or Control and Systems Engineering and indexed in Web of Science. For conferences papers, instead, we referred to the CORE[3] conferences ranking system, which provides an assessment of the major conferences in the field, excluding papers published in venues ranked B or below (i.e. we keep only A* and A – the top 22%). We then refined the bibliographic search by manually checking for other relevant papers exploiting suggested similar publication in Google Scholar (still matching the above constraints). This process yielded the selection revised in this survey.

### C. HOW TO NAVIGATE THIS SURVEY

This paper is organised as follows. In Section II, we provide a concise review of the theoretical background behind graph learning, with specific focus on different graph convolution formulations. Following, we explore the use of GNNs in robotics applications. First, in Section III we showcase the application of GNNs in the context of single-agent settings. With such definition we refer to cases where graphs are used to model one single item, which encompasses three main cases: modelling of objects (Section IIIa), modelling of hands-objects interaction (section IIIb) and modelling of human behaviour (Section IIIc). Then, Section IV discusses

[1] https://www.scopus.com
[2] https://www.scimagojr.com
[3] http://portal.core.edu.au/conf-ranks/

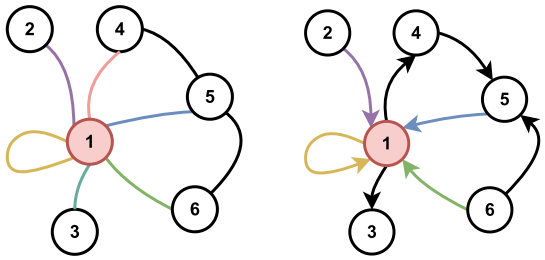



**FIGURE 2.** Undirected (left) and a directed (right) graph representation. The neighbourhood of node 1 in highlighted through coloured edges in both graphs. For the undirected graph, the neighbourhood is composed by the nodes that are connected to node 1: node 2, node 3, node 4, node 5, node 6. Instead, for the directed one, the neighbourhood is composed only by the nodes that have incoming edges to node 1: node 2, node 5, node 6. It is worth mentioning that in some cases graphs may also have edges starting and ending on the same node (self-loop), which may be relevant for some graph convolution formalisation.

multi-agent settings, where graphs are used to model the interaction between multiple items. The section is further divided in Task and motion planning (Section IVa) and multi-robot exploration and navigation (Section IVb). We also provide a section where we discuss strengths and limitations of current approaches including GNNs in their pipeline, and finally a conclusion with a forecast of future perspectives. All the parts are self-contained, allowing the curious reader to directly refer to a specific section of interest.

## II. PRELIMINARIES ON GRAPH NEURAL NETWORKS

In the last few years, learning on graph structures has gained significant relevance in the community, thanks to their intrinsic capability to easily process data lying on irregular domains, such as three-dimensional point clouds, spatio-temporal and functional relationships [14], [15], [16], [17]. Similarly to what Convolutional Neural Networks (CNNs) did for standard images, the community has devoted significant effort in defining convolution-based aggregation mechanisms for graph structures.

A graph convolution operation can be formulated over two different domains, spectral or spatial. The first family of methods [18], [19], [20], [21], [22] usually exploits graph Fourier transform, eventually complemented with polynomial approximations to reduce the computational burden [20], [21]. Among these, it is worth mentioning the Graph Convolutional Network (GCN) [21], which demonstrated notable results for semi-supervised problems, such as semi-supervised node or multi-class classification. However, a critical limitation of this formulation is the inability to generalise the learned filters, computed over the spectrum of the graph Laplacian, to a variable graph structure.

The second class of approaches defines the graph-convolution operation in the spatial domain. In this scenario, the graph convolution is defined as a local, i.e. computed over a neighbourhood, weighted aggregation of signals. Since it is defined at the neighbourhood level, this formulation is suitable for any type of signal that can be defined over a graph, even with a variable graph structure. Several definitions are present in the literature [5], [6], [24], [25], [26], [27], [28], [29], [30]. These often differ on the computation of the weights used in the aggregation. For example, many formulations use scalar weights [24], [25], [27] or matrices [6], [26], [28], [30] independent from the input data. On the other side, [5] proposes edge-dependent matrices as weights, providing an operation with more representational power. For a compact overview, the reader could refer to Fig. 3 for different graph convolution formulations and to Tab. 1 for a comparison between implementations.

### A. GRAPH SIGNAL THEORY BACKGROUNDS

Given a graph $G = \{V, E\}$, we refer to $V$ as the set of nodes, with cardinality $M = |V|$, and $E$ as the set of edges. Two nodes, $n_i$ and $n_j$ are said to be connected only if there exists an edge $e_{i,j} \in E$ between them. The graph is also undirected if $e_{i,j} = e_{j,i}, \quad \forall e_{i,j} \in E$, otherwise the order of the indices reflects the direction of the graph and in general $e_{i,j} \neq e_{j,i}$. Each node or edge can be associated with a feature vector, $F_v \in \mathbb{R}^{d_v}$ or $F_e \in \mathbb{R}^{d_e}$. The neighbourhood of the $i^{th}$ node is defined as $N(i) = \{j | e_{i,j} \in E\}$, and corresponds to the set of nodes directly connected to the $i^{th}$ node. In Fig. 2 it is reported an example of an undirected and a directed graphs, where the neighbourhood of a node is highlighted.

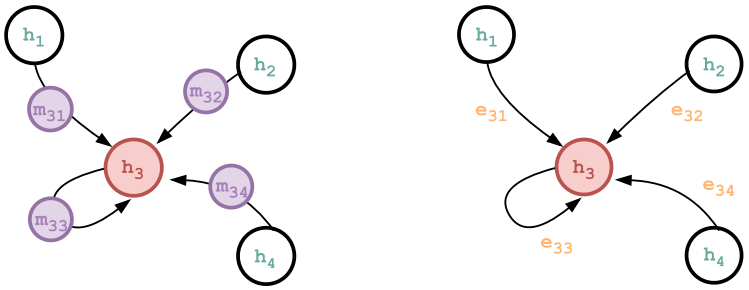
The connectivity of the graph can be represented through the adjacency matrix $A \in \mathbb{R}^{M,M}$, where element $A_{i,j} = 1$ if exists the corresponding edge $e_{i,j} \in E$, and zero otherwise. Note that, if the graph is undirected, the corresponding adjacency matrix is symmetric. From the connectivity matrix it is possible to define the diagonal degree matrix $\mathbf{D}$, where the diagonal element $d_{i,i}$ is equal to the sum of all the edges, incoming and outgoing if in presence of a directed graph, relative to the $i^{th}$ node. In some cases, edges could be weighted by specific values $w_{i,j}$, and the corresponding element of the adjacency matrix becomes $A_{i,j} = w_{i,j}$ if $e_{i,j} \in E$, and zero otherwise.

Finally, it is worth recalling the definition - for undirected graphs - of the Laplacian matrix $\hat{L} = D - A$, and of its normalised formulation $L = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$. Since the normalised graph Laplacian is a semi-define positive matrix, it can be decomposed as $L = U \Lambda U^T$. Given a generic graph signal $f : V \to \mathbb{R}$, which can be represented in vector form as $x \in \mathbb{R}^M$ where each element $x_i$ is the function evaluated in the $i^{th}$ node, it is possible to define the graph Fourier transform $\mathcal{F}$ of the signal $\mathbf{x}$ and its inverse $\mathcal{F}^{-1}$ as:

$$\mathcal{F}(x) = \hat{x} = U^T x = \sum_{i=0}^{M-1} x_i u_i, \tag{1}$$

$$\mathcal{F}^{-1}(\hat{x}) = x = Ux = \sum_{i=0}^{M-1} \hat{x}_i u_i. \tag{2}$$

The graph Fourier transform projects $\mathbf{x}$ into an orthonormal space where the eigenvectors of the graph Laplacian matrix are the basis of the new space. Thanks to their definition,




$$m_{ij}^{MPNN} = f_e(h_i, h_j, e_{ij}) \qquad e_{ij}^{GCN} = \frac{1}{\sqrt{|N(i)|}\sqrt{|N(j)|}} \qquad e_{ij}^{ECC} = MLP(h_i - h_j)$$
$$\hat{h}_i = f_v(h_i, \sum_{j \in N(i)} m_{ij}) \qquad e_{ij}^{GAT} = softmax_j(a(\mathbf{W}h_i, \mathbf{W}h_j)) \qquad \hat{h}_i = \frac{1}{|N(i)|} \sum_{j \in N(i)} e_{ij} h_j$$
$$\hat{h}_i = \sigma(\sum_{j \in N(i)} e_{ij} W h_j) \qquad e_{ij}^{DGCNN} = ReLU(\theta(h_i - h_j) + \phi(h_i))$$
$$\hat{h}_i = max(e_{ij})$$

**FIGURE 3.** Graph convolutional operations with different information sharing mechanism. MPNN [23] implements message passing conditioned by edge attributes. GCN [21] is reported in its equivalent spatial formulation where it implements a simple aggregation of the neighbouring nodes. With a similar operation, GAT [24] exploits self-attention to weight the contributions of the nearest neighbours. Instead ECC [5] implements an edge-depended weighting function to weight the contribution of each node in the neighbourhood. DGCNN [6] aggregate the information of the neighbourhood by means of learned edge features.

the graph Laplacian operators family captures the differences between a signal at a node and its surroundings, and specifically the graph Laplacian is often used as a measure of signal smoothness. Using the graph Fourier transform, it is possible to filter the signal either in the spectral or in the spatial domain, and define a graph convolutional operation.

### 1) SPECTRAL GRAPH CONVOLUTIONS

In the spectral domain, the filtering operation corresponds to a multiplication between the graph signal $x \in \mathbb{R}^M$ and the filter $g \in \mathbb{R}^M$:

$$x *_G g = \mathcal{F}^{-1}(\mathcal{F}(x) \odot \mathcal{F}(g)) = U((U^T x) \odot (U^T g)), \quad (3)$$

where $\odot$ is the Hadamard (element-wise) product.

Considering a filter in the form $g_\theta = diag(U^T g)$, the spectral graph convolution can be simplified to:

$$x *_G g = U g_\theta U^T x. \quad (4)$$

The distinct definitions of spectral graph convolution differs for the choice of the filter $g_\theta$. Bruna et al. [18] proposes a first repurpose of classic CNNs to the graph domain, and introduces a spectral convolution where the graph convolutional filter is modelled as a learnable diagonal matrix $g_\theta = \Theta$ obtaining a simple yet effective formulation. Defferrard et al. [20], instead, proposes an efficient formulation approximating the filters as Chebyshev polynomials $g_\theta = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L})$. $T_k(\tilde{L})$ is the Chebychev polynomial of order $k$ evaluated at the scaled Laplacian $\tilde{L} = \frac{2L}{\lambda_{max}} - I$ and $\theta$ is a learnable vector of length $k$. Recall that the Chebychev polynomials are recursively defined as $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ with $T_0(x) = 1$ and $T_1(x) = x$. Given an input signal $x^t$ at step $t$, the output signal $x^{t+1}$ after the graph convolution is:

$$x^{t+1} = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L}) x^t. \quad (5)$$

An interesting observation is that the filters are localised in a spatial neighbourhood. Such formulation is further simplified by Kipf and Welling [21], which proposes a first-order approximation of the Chebychev polynomials and exploits the normalised graph Laplacian build upon the adjacency matrix with self-loop connections $\hat{A}$ instead of the scaled Laplacian:

$$x^{t+1} = \alpha(I - L)x^t, \quad (6)$$

where $\alpha$ is a learnable scalar parameter. Such formulation consists in simply taking a weighted aggregation of the signals in the neighbourhood of each points with weights proportional to the connectivity matrix.

Such models have the advantage of providing a formulation for filters localised in the neighbourhood space, with a limited number of learnable parameters and independent from the graph size. Nevertheless, they are linked to a fixed graph structure, given by the graph Laplacian, and therefore are not able to generalise to other configurations. Furthermore, they are not robust to perturbations that would change the basis functions, and they require a computational cost of $O(N^2)$.

### 2) SPATIAL GRAPH CONVOLUTIONS

As discussed before, an alternative representation of the graph convolution operation may be also formulated in the spatial domain. The filtered signal $x_i$ at the $i^{th}$ node is computed as the linear combination of the signal itself and its k-hop local neighbourhood $N_k(i)$. The k-hop neighbourhood of node $i$ is

**TABLE 1. Comparison between different Graph Convolutional Neural Architectures. For each implementation, we report few core characteristics: the type (spectral vs. spatial), how weights are computed, the availability of attributes for edges and a list of relevant papers where these have been used.**

| Name | Type | Weights computation | Edges | Robotic Applications |
|---|---|---|---|---|
| Bruna et al. [18] | Spectral | lernable diagonal matrix | w/o attributes | - |
| Defferard et al. [20] | Spectral | learnable parameters | w/o attributes | [31] [12] |
| GCN [21] | Spectral | learnable parameters $\propto$ connectivity | w/o attributes | [32] [9] [33] [34] |
| GNN [3] | Spatial | learnble paramters | w/o attributes | [35] [36] [37] [38] [39] |
| MPNN [23] | Spatial | learnable matrices | w/ attributes | - |
| GatedGCNN [40] | Spatial | lernable matrices | support multiple types | [41] |
| Monti et al. [25] | Spatial | edge weighting function | w/ attributes | - |
| GraphSAGE [26] | Spatial | MLP | w/o attributes | - |
| GAT [24] | Spatial | self-attention | w/o attributes | [42] [43] |
| ECC [5] | Spatial | edge weighting MLP | w/ attributes | - |
| DGCNN [6] | Spatial | MLP | w/ attributes | - |

defined as the collection of nodes that have the shortest path distance from the $i^{th}$ node less or equal to k. In this survey, we assume the standard practice of considering the 1-hop neighbourhood of a node - if not specified otherwise - and $N_1(i) = N(i)$. Therefore, it is possible to define the spatial graph filtering as:

$$x_i^{t+1} = W_{i,i}x_i^t + \sum_{j \in N_k(i)} W_{i,j}x_j^t, \tag{7}$$

where the first part refers to the contribution of the $i^{th}$ node itself, also called self-loop, and the second to the contribution of the surroundings points. The spatial approach shares similar principle ideas with propagation and message passing of recurrent neural networks, since they both propagate and exchange information within the neighbourhood structure. One of the first message passing models is presented in [3], where the signal function at node $i$ is updated at each time step $t$ with the aggregation of the information, also called messages, from each node $j$ in its neighbourhood. It implements an extension of previous recurrent graph neural networks to more general scenarios, considering different types of graphs. The proposed message passing mechanism can be formulated as follow:

$$x_i^{t+1} = u(F_{v,i}, \sum_{j \in N(i)} m(x_i^t, F_{v,i}, F_{v,j}, F_{e,(i,j)})), \tag{8}$$

where $m$ is the message passing function, $u$ the update function, $F_{v,i}$ the feature vector associated to the $i^{th}$ node and $F_{e,(i,j)}$ the feature vector associated to the edge that connects nodes $i$ and $j$.

Over the years, several models based on message passing have been developed [1], [23], [44], [45], where different message passing and update functions are proposed. Among the others, it is worth mentioning Gilmer et al. [23], which proposes a general model called Message Passing Neural Network (MPNN), that implements a spatial graph convolution by means of message passing, where the information is shared between nodes conditioned by the edge attribute. Instead, the GatedGCN proposed in [40] includes a Gated Recurrent Unit (GRU) [46] and the support for different edge types and directions. Similarly, Monti et al. [25] presents a Gaussian mixture model that exploits continuous edge attributes.

A spatial graph convolutional neural network can be interpreted as a local weighted aggregation of points. Locality reflects the computation performed only at neighbourhood level, while weights are used to scale different contributions. The aggregation function should keep the permutation invariance to the node ordering, and therefore it is usually a sum, mean or max function. In the literature, researchers proposed several spatial graph convolutional formulations which differ in one of the three main characteristics of the function: computation of the neighbourhood, weighting function and type of aggregation. GraphSAGE [26] proposes to sub-sample the neighbours, aggregate their information with the centre node and then project into the feature space, promoting similar embedding in close nodes. Graph attention networks (GAT) [24], instead, exploits a self-attention mechanism to compute the weight for each neighbours. Edge-Conditioned Convolution (ECC) [5] proposes an edge-dependent convolution definition. In particular, the neighbouring contributions are weighted by the output of a multi-layer perceptron that takes as input the corresponding edge labels, commonly defined as the difference between the features associated to each neighbours and the centre point. Instead Dynamic Graph CNN (DGCNN) [6] proposes a novel formulation called EdgeConv to generate edge features to describe the relationships between a centre point $i$ and its nearest neighbours. The edge features are defined as the output of a multi-layer perceptron that takes as input the features associated to each node $j \in N(i)$ and the centre node $i$ and are directly aggregated together to obtain the new feature vector of the centre node. Remarkably, DGCNN [6] first introduces the concept of dynamic graph construction, i.e. the edges and consequently the neighbourhood of each node is updated after several graph convolutional layers.

As we can observe from the previous description, one important difference between graph convolution formulations is the contribution of the neighbourhood: GraphSAGE [26] considers each contribution equal, GCN [21] uses a fixed

defined a-priori weight per neighbour, while ECC [5] proposes edge dependent weighting contributions.

## III. GNNS FOR SINGLE-AGENT SYSTEMS

In this survey, we refer to ''Single-Agent Systems'' either when we review papers dealing with only one intelligent system (single-robot or human), or passive bodies (objects modelling). All the above scenarios share the common principle of using graphs to encode or guide the action of a single robot. For the sake of clarity, we further split the topic in: i) modelling of bodies, where graphs are used to encode the dynamic or kinematic behaviour of a passive object (usually compliant, such as in [35] and [47]); ii) modelling of robots interacting with objects, mostly for grasping and manipulation purposes (e.g. [32]); iii) modelling of human behaviour and human-centric environments (e.g. [9]).

### A. MODELLING OF BODIES

When we interact with a compliant object, for instance during grasping or manipulation, very likely it will suffer a non-linear deformation which depends on the force and torques applied by fingers and the external environment, and on the dynamic behaviour of the body itself. However, modelling how objects deform when subject to external forces is an open problem, which poses several technical and theoretical challenges, such as: i) the high dimensionality of the configuration space, ii) the non-linear dynamics of deformable materials, and iii) possible self-occlusion in visual perception.

One promising approach relies on the discretisation of the continuous body representation, through the identification of a finite number of keypoints. The spatial and kinetic relationship between keypoints can be easily represented through edges connecting neighbouring keypoints. Such graph encoding can be used to provide a compact, yet approximated, representation of the body shape and, potentially, its dynamic behaviour [2], [47]. Message passing between nodes of a graph is a powerful tool able to learn useful spatio-temporal information on the interaction between elemental components of compliant bodies. Recently, learning control policies from deformable objects keypoints has gained attention [31], [47], [48], [49], [50], [51] thanks to their effective low-dimensional alternative representation.

One of the first works going in this direction is proposed in [47], where the authors move from the observation that particle-based simulators [52] are widely used to model complex dynamics, but very often imperfect because of a set of approximations which does not scale well in real-world scenarios. To tackle this problem, in [47] the authors proposed to learn a particle-based simulator with a network able to model the dynamic interaction between particles of an object through a graph neural network. This approach comes with the relevant benefit of being suitable for a wide range of objects (rigid, soft and fluids), and demonstrated interesting capabilities in inferring an inductive bias on the type of particles, useful to quickly adapt to unknown environments. The proposed interaction model takes inspiration from [53], extending it to particle-based dynamics. Interestingly, the authors introduce a dynamic graph construction, where nodes are all the particles of an object, and edges - representing the interaction between particles - are dynamically updated in time to grant a trade-off between efficiency and effectiveness. This is an interesting design choice which - compared to the alternative fixed fully connected graph - has the advantages of being more efficient and more effective. In real-world physical systems, since not all the points would interact with all the others, it is reasonable to assume a distance-based neighbourhood, and they involve discontinuous functions that can not be taken into account with a fixed graph construction. They also introduce a message-passing mechanism [54] which is object-type dependent, meaning that different rules are applied for rigid bodies, deformable objects, and fluids. The same dependencies are applied to the graph construction, leading to different graphs for different types of objects.

Although particle-based approaches demonstrated interesting results in terms of accuracy, in many cases they also require a large number of particles to properly approximate a realistic object, which may result in excessive computational cost, prohibitive for real-time scenarios. Alternative solutions, which mitigate this issue, usually approximate the overall complexity by focusing only on a strategic subset of points, also known as keypoints, easy to detect and sufficient to estimate the object dynamics. As an example, G-DOOM [48] extracts keypoints from depth images via unsupervised learning and uses such data as nodes of a graph. A graph convolutional neural network is then used to capture the underlying geometry and abstract interactions between keypoints. They also introduce a recurrent structure in the architecture to track keypoints over time, with the aim of handling potential keypoints occlusions.

The findings discussed above suggest that keypoints-based discretisation seems to be the right choice to enable a low-latency simulation of soft bodies. However, being an approximation of the real world, it is worth remarking that this will also introduce a non-negligible discrepancy between simulated and real-world data distribution (sim-to-real gap) which, in some cases, may produce important deviations from the simulated environment. To address this issue, [49] applies GNNs in an offline-online framework in an attempt to mitigate the sim-to-real gap, proving the capability of GNNs to generalise well across data distribution and to fit global models. In the context of a robot tasked to re-arrange the shape of a compliant linear object (cable-like), the authors proposed a two-stage learning process. In an offline phase, a GNNs is used to learn the deformation dynamics of the cable purely from simulated data. Then, in the online phase, a linear residual model is learned to minimise the sim-to-real gap between a simulated and the real cable. The trained model is then used to constrain the dynamics of a trust-region -based Model Predictive Controller (MPC) [55], which computes the optimal robot motion for cable rearrangement.

Another challenge that may arise when using particle-based models lies on the fact that these often require a temporal consistency between particles. Focusing on the modelling of elasto-plastic objects, in [50] the authors propose to overcome this limitation by training a GNN to model the object dynamics with distance-based supervision between particle distribution, and to predict the deformation of objects when subject to external wrenches (e.g. applied by grippers). Interestingly, this representation can be applied in many complex scenarios, such as goal-conditioned rearrangement tasks [31], [51], where graph-based architectures can be used to directly learn manipulation policies from keypoints, taking as input the keypoints of the current and target state and learning to predict the optimal pick and place set of actions to re-shape the object toward the target configuration. More specifically, two initial sets of detected keypoints are used as nodes of two graphs, encoding initial and target configuration. These are then fed into a local-GNN which implements self-attention operations within each graph, and cross-attention operations across graphs, to perform keypoint matching. These works provide evidence that the use of graphs allows the method to efficiently model the high-dimensional deformable configuration space and its underlying complexity, nonlinearity, and uncertainty, which are intrinsic in deformable object dynamics.

Particles and keypoints are often inferred from visual representation, such as point clouds or mesh [50], images [31], [51] and depth images [48]. Recently, directly using mesh representations has gained attention as an alternative approach to model complex systems, especially for deformable objects [35], [36], [56], [57]. Among the others, it is worth discussing firstly MeshGraphNets [35], a general framework able to learn the dynamics of a wide range of systems, from cloths to fluids. In a nutshell, the authors propose to encode the state of a mesh, obtained in simulation, in a graph structure that includes mesh-space nodes (i.e. nodes associated with the object) and world-space nodes (i.e. nodes with extra edges representing interaction with external objects and environment). The mesh is processed through an Encode-Process-Decode architecture where the encoder is responsible to build the graph mentioned above, the processor performs message passing along mesh edges and world edges to update nodes embeddings, and the decoder finally computes nodes acceleration (which iterated will produce motion). Interestingly, the authors argue how message passing in mesh-space makes the model learn the internal dynamics of the physical system, while message passing in world-space captures external dynamics such as contacts. Thanks to this graph-based encoding, MeshGraphNets demonstrated notable capabilities in learning resolution-independent dynamics, enabling variable resolution at runtime, and even an adaptive change of discretization during rollouts, opening to the interesting possibility of allocating larger computational resources to specific local regions where higher accuracy is needed (e.g. corners or contact points).

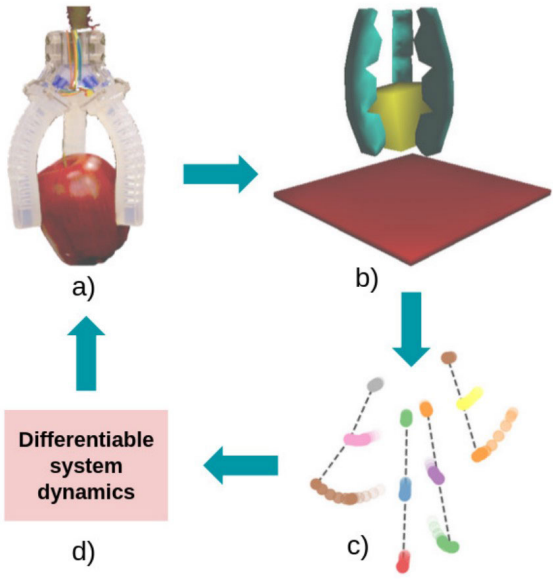



**FIGURE 4.** An example of graph used to model a soft end-effector. The GNN is used to learn the dynamics of the system by tracking a selection of keypoints on the robot body. Image taken from [58]. Copyright ©2021, IEEE.

In a similar fashion, but in the context of manipulation planning, DefGraspNets [36] propose a multigraph to encode the object mesh together with the gripper mesh, which is then used to learn to predict the deformation of soft objects during manipulation.

One of the main challenges that arise when learning robotics tasks that imply interaction with deformable objects, such as cloth manipulation, is the intrinsic difficulty in retrieving the object connectivity, because of the large number of degrees of freedom and the self-occlusion conditions. Graphs can be used to tackle this problem and extract directly from visual data the connectivity of a cloth [56], [57]. These methods usually move from a graph built upon the voxelization of an object point clouds, where each node of the graph is the centroid of the corresponding voxel cell, and nodes that are geometrically close to each other (but not necessarily physically) are connected together. A GNN can be used to estimate whether each edge of the graph is also a mesh edge, meaning that it represents also a physical connection of the cloth structure. Then, this visual connectivity graph can be used to learn the model dynamics by means of a GNN that estimates the acceleration of each particle (i.e. of each node).

Graphs can be used not only to model deformable objects but also the robot itself. In general, human, humanoid or animal bodies have a discrete graph structure, where nodes are joints and edges their physical dependencies [37], [59]. NerveNet [37] builds a graph to represent the structure of the agent, where nodes are different parts of the robot and edges

are weighted proportionally to the distance between nodes, and learns continuous agent's control policy by means of a graph neural network, using the implementation of [3]. The state of each node is updated based on both the aggregated message from its neighbourhood and its current state vector. Such structure has several advantages, the resulting learned policy is general and zero-shot transferable to new scenarios.

In [59], instead, the authors use a similar model where visual observations are inserted as additional inputs, leading to a graph that encodes both agent and environment status. Both these works prove the importance of training a policy that is aware of the kinematics of the robot. Particularly challenging is the modelling of soft robots, due to their complex continuous dynamics [60]. Related to this topic, it is relevant to mention [58] (see Fig. 4) where the authors demonstrate how building a graph upon keypoints to model the agent, and using a graph neural architecture, may enable the model to learn to exploit the underlying physical structure.

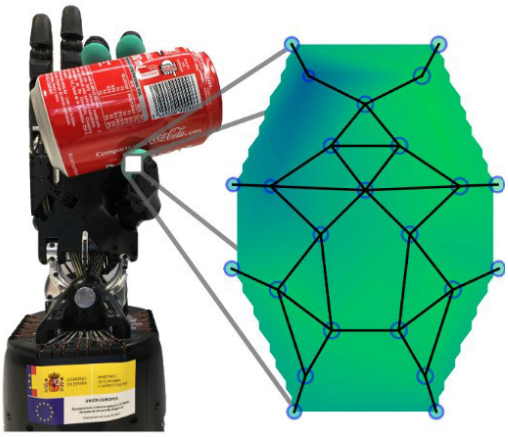


**FIGURE 5.** Shadow Robot Dexterous robotic hand endowed with a BioTac SP tactile sensors. Each sensor is represented as graph encoding the continuous sensing surface. Image from [83]. Copyright ©2019, IEEE.

### B. GRASPING AND MANIPULATION

One of the primary tasks that an intelligent robot should be able to perform is to safely and effectively interact with objects in its workspace. Indeed, many practical applications require to move, manipulate and use a large variety of objects and tools with different sizes and shapes.

The capability to firmly grasp an object, or to apply controlled wrenches on it with the purpose of changing it pose (manipulation) is one of the primary skills we envision in home-integrated robotic devices. However, although at a first glance this may seem as an easy problem, matching human manipulation skills with modern manipulators is incredibly challenging and a multifaceted problem [61].

The development of fully autonomous manipulation systems requires targeting at least three main issues: perception, planning, and control. The first refers to the problem of making the robot able to perceive and understand the surrounding scene, identify where (and which) objects are available in the workspace, and how the environment is composed (e.g. static and dynamic obstacles). At this stage, it is particularly important to endow the robot with the capability to learn a representation of the environment and of the objects, which can be used to inform the robot actions for task execution.

Then, once the robot is aware of the scene and of the objects that will be asked to grasp or manipulate, it will be relevant to plan an accurate end-effector motion, which requires the availability of models, usually probabilistic, that predict object state changes when subject to robot actions.

Lastly, the planned trajectory will be executed with a dedicated controller, which needs to be sufficiently robust to account for disturbances in sensing and for errors in planning and perception, yet at the same time adequately compliant to avoid harsh interactions and damages.

Albeit being a research problem for decades [62], [63], in robotic grasping and manipulation several problems are still open, such as learning to interact with objects in motion or in clutter [64], [65], [66], manipulation of deformable objects [67], [68], [69], [70], [71] - for which a proper modelling is crucial as discussed in Sec. III-A - and human-robot or multi-robot co-manipulation [72], [73], [74], [75]. In addition, when it comes to the deployment of these systems in real-world settings, we may also encounter additional issues: occluded objects due to cluttered scenes or partial observability, novel objects, characterised by new shapes or dimensions, or more sophisticated grasping strategies, for example task-conditioned, where the goal is to select the grasp pose and location depending on the task for which the grasp is performed (i.e. grasping a knife will be executed differently depending whether the robot will cut something, or just handout the tool to someone).

In the last few years, learning for graphs has provided interesting results in tackling many open problems in this field. Graphs can efficiently model the environment and the interaction between objects [76], [77], [154], [155], learn semantic global information to build knowledge graphs [32], [78], [79] or process unstructured data input [80], [81], [82].

Graph-based structures have been also profitably used to encode spatial relations in complex scenes, where multiple objects are present and/or the environment is only partially observable [41], [77], [84], [85], [86], [87], [88], [89], [90]. Object-oriented scenes can be encoded in a graph where nodes are objects and the edges reflect the relations between them, providing a powerful and comprehensive representation of the environment. Notably, since graphs are invariant to the number of nodes, such representation is invariant with respect to the number of objects and therefore can be applied to generic scenes with an arbitrary number of objects.

In real-world scenarios, it is quite common to have cluttered scenes where it is required to perform grasping minimising the probability to cause collisions. The majority of methods usually solve this problem in two steps, first predicting the grasping pose of an object, and then using a

collision checking module to verify the safety and feasibility of the planned graph [91], [92], [93], [94]. In many cases, however, objects are processed independently from the surrounding scene, and the spatial relationship between items is usually disregarded when predicting the grasp pose, which may lead to sub-optimal results and errors in the presence of partial observation. Yet, several papers demonstrated how building a graph of the objects in the scene enables the exploitation of objects' surroundings knowledge, useful to directly predict effective and safe grasps [41], [85], [86]. Graph-based encoding showed also interesting features in handling multiple instances of the same item. Assuming to have several similar (or identical) samples of the same object - e.g. in tool boxes, shop carts, or groceries shelves - GNNs are able to capture and highlight the hidden correlation between nodes, which may be exploited for instance to select the most accessible sample of the desired object [85]. In a similar fashion, [84] tackled the problem of grasping partially visible objects. A graph conv in this case is used to uncover and explore non-observable parts of the scene, and to aggregate information extracted from different regions, to capture internal spatial relations and decide whether the target object is already accessible or requires a non-prehensile manipulation primitive (e.g. push) to rearrange its pose.

A relevant topic worth discussing is also the analysis of dynamic multi-object interactions. References [77], [86], and [87] examine long-horizon planning task, where several consequent actions are planned to achieve the desired goal. Reference [86] proposes a graph neural network to learn manipulation effects on multiple objects, and proves that the relational inductive bias of this type of network is effective in planning even on a very long time span. Similarly, [41] uses the graph relations between objects to predict the grasping order of multiple objects. In [77] long-term manipulation tasks are seen as a sequence of spatial constraints and objects relationship. The authors proposed to represent the environment state as a graph, where each node encodes features associated with objects and their goals (i.e. target positions). A graph-based architecture is then trained to select the next object to move, where to place it, and predict a task-specific action. Interestingly, they also propose an expert demonstration cast as a graph, named GNNExplainer. Their work suggests that graphs are good to encode the knowledge hidden in the examples provided in supervision and not encoded in the problem. This powerful representation of the environment, able to capture spatial dependencies, can be exploited in other more uncommon tasks, as done by [88] where the goal was to learn how to place objects in a scene following user's preferences. Similarly, graphs can also be used to model the interaction between objects and the robot in the context of bi-manual robotic manipulation [76]. In this work, the authors introduce a two-level framework composed of a decomposition of the multi-modal dynamics into primitives, and a primitive dynamics model where graph recurrent neural networks are used to capture interactions between objects and the robot arms. In particular, with the graph they model the interaction between different parts of the robot and the manipulated objects. In this case, graph representation helps to improve the performance of the model in several simulated bi-manual robotic manipulation tasks, showing interesting capabilities in increasing training speed and overall accuracy.

So far we mostly discussed a selection of methods where graphs demonstrated to be convenient tools for modelling environment-level spatial relations between objects and robots. Interestingly, the same spatial representation problem can be casted to a smaller scale, and model the relationship between fingers and fingertips contacts during grasping (see e.g. Fig. 5). Many works, indeed, propose to leverage tactile sensing for the assessment of grasp stability or to foster effective in-hand manipulation [95], [96], [97], [98], [99]. However, although in most cases these sensing strategies leverage cameras to measure sensing surface deformations, their output cannot be easily processed by standard convolutional architectures, because of the complex non-linear relationship between different sensing units. Notably, these issues are discussed and addressed for example in [83] for tactile sensor state estimation (single sensing unit) and in [100] for multi-finger pose estimation (multiple sensing unit). The construction of the graph from tactile sensors of [83] is reported in Fig. 5 as an example.

An insightful application of graphs in the context of robotic grasping and manipulation is the creation of a knowledge graph [32], [78], [79], [101], [102], [103], [104], [105], where semantic knowledge and its internal relations are organised in an efficient and powerful way.

This structure is able to capture the semantic knowledge about objects, tasks, agents, actions, and regions present in the training environments, and potentially to generalise to novel scenarios, types of objects, shapes, and tasks. Large-scale knowledge graphs such as RoboBrain [102] and RoboCSE [103] propose general frameworks able to extract abstract semantics concepts that can be generalised and take into account the uncertainty of real-world scenarios. Such structure can be exploited for sophisticated robot manipulation, such as for task-oriented grasping first proposed in [32], where the authors present a framework, named GCNGrasp, where a knowledge graph that encodes objects, tasks and object hierarchies from WordNet [106] is exploited to decide whether a test grasp pose is suitable for the desired task on that specific object (see Fig. 6). The classification is done by adding the query grasp pose node on the knowledge graph and by performing few graph convolution layers. Instead, [78] builds a graph, called RoboKG, with all the available information of the objects, such as material and components, of the tasks, and of the manipulation characteristics, such as type of gripper and exerted force. RoboKG is then used to infer which gripper to use, which part of the object to grasp, and the amount of force required.

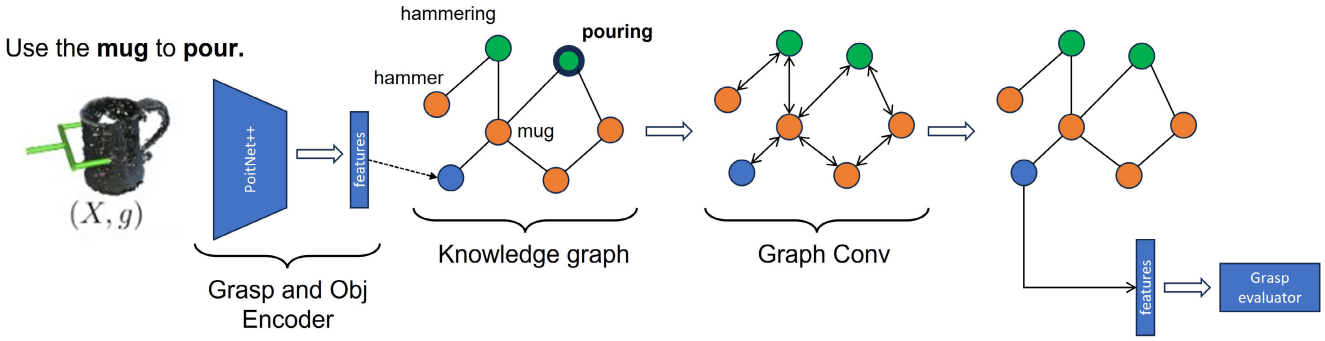



**FIGURE 6.** Overview of the method proposed by [32] for task-oriented grasping. It is interesting to observe the utilisation of the knowledge graph to infer information about the query grasp pose. Image redrawn from [32].

Graphs can also be used to encode the functional relationships between grasping strategies, providing a way to contaminate classes representation and to generalise to novel grasping skills. In [79] the authors propose to generate human-inspired grasping strategies depending on objects shapes, claiming that the object shape provide a good affordance for the grasping type, e.g. a cup would be more likely picked with a cylindrical grasping, while a pencil would require a precision pinch grasp. One of the main problems of this representation is the unbalanced dataset, as some strategies are much more rare than others. To overcome this issue, a knowledge graph, built with the knowledge of a kinematic relationship between classes, is used to transfer information between nodes and improve prototypes of long-tail classes.

In robot manipulation, and especially grasping, it is more and more popular to exploit 3D input data, since the depth information is important to a proper grasp prediction. However, handling 3D data representation such as point clouds is significantly harder than common RGB images, since they lie on non-Euclidean domains, and classic neural network architectures - such CNNs or MLPs - are not effective or even applicable. Graph neural networks have emerged as efficient methods to deal with unstructured data, as demonstrated by a wide literature proposing graph-based architectures to process point clouds [5], [6], [81]. Graph representation of 3D data has already been used in robotics with a straightforward implementation of standard GNNs. In [80] the authors propose a method to improve task-dependent grasping by finding the object category from a given ontology. The authors pose an interesting assumption on the fact that objects with similar shapes would be grasped in a similar way for similar tasks. Thanks to the representation of each 3D object as a graph, where nodes are points and edges are weighted by the changes of the surface, graph kernels are used to compare the similarity between objects, with the purpose of fostering knowledge sharing between objects of similar shape.

### C. HUMAN ACTION RECOGNITION

Human action recognition, tracking, and forecasting are rapidly becoming a key enabling factor for a variety of robotics applications, ranging from technology-enabled healthcare, all the way to assistive robotics and human-machine interaction and cooperation. As an example, it is worth mentioning gesture recognition, where the robot is tasked to detect and interpret human hand and arm movements, to favour a natural interaction [107]. This can be used in a wide variety of different ways, for example, a robot in a healthcare facility could be able to recognise specific hand gestures made by patients or doctors, and therefore trigger specific reactions, e.g. fetch tools or program actions [108]. Human gesture understanding can be also used to feed learning-by-demonstration or imitation learning pipelines [109], [110], [111], with the goal of transferring human skills to manipulators, making them able for example to fold a towel [112], paint a piece of furniture or manipulate kitchen items [113].

Beyond making robots more and more autonomous for daily living tasks, an open and - in some cases - more challenging problem is to enable a safe physical interaction with the environment and the humans [114], [115]. In this case, human action recognition becomes even more crucial, because the robot may benefit from understanding whether people in its surrounding will walk or move their limbs in specific portions of the workspace, to either avoid contacts or trigger consequent support actions (e.g. pass a tool or co-manipulate an object) [116], [117], [118]. Of note, understanding human behaviour, and eventually being able to react consequently, may have a significant effect on the anthropomorphism and accountability of the machine, which proved to be a key factor for the increased acceptability and trustworthy of machines in our daily life [119].

The use of graph neural networks for human motion modelling has a longstanding tradition, especially for body keypoints estimation [120], [121], [122], [123]. Several surveys have been proposed to summarise the state of the art of deep learning methods for skeleton-based human pose reconstruction (as for example [124]), and therefore will be here omitted to preserve the robotics-oriented focus of this paper and to leave more space to the modelling of complex human behaviour when interacting with objects and the surrounding environment.

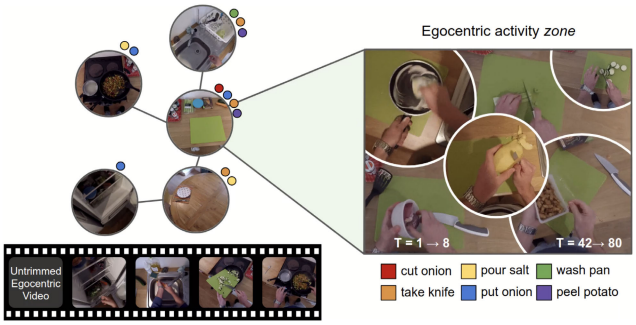



**FIGURE 7.** Topological graph of activity-centric zones. Image from [9]. Copyright ©2020, IEEE.

Human behaviour is an intricate and multifaceted phenomenon, strongly affected by individual personality traits, past experiences, cultural background, and social influence, which are interleaved in a unique and unpredictable way. These aspects play a role even in controlled scenarios, e.g. during a meal preparation we may choose different steps order depending on our habits. To master this complexity, graphs appear to be a valid choice, thanks to the possibility to represent in a compact fashion the functional or semantic relationship between knowledge entities encoded in nodes. Under the domain of vision-based systems, which represents the primary sensing modality for this kind of application, graphs can be used to model structured information such as zones of an environment (the sink, the oven and the fridge in a kitchen, the desk, the armchair and the window in an office), or contacts between hands and objects [125], [126]. This line of research has been particularly fostered by the release of large unstructured datasets, among which it is worth mentioning Kinetics [127] and Charades [8] for third-person videos, and Epic Kitchens [128], Ego-4D [129], and Assembly101 [130] for egocentric vision.

One notable example is [9], where the authors discuss the feasibility to detect relevant zones in a domestic environment from egocentric videos (activity-centric zones) and encode them in a graph structure where new zones explored are sequentially added to the graph. Interestingly, such topological representation of environmental key zones, reported for reader's convenience in Fig. 7, intrinsically embeds information on which actions may take place in specific zones (i.e. if the camera is looking at a sink, the user will likely wash something), while edges capture spatial relationships between different zones, depending on how people navigate between them. Topological graphs like the one proposed in [9] demonstrate an interesting capability to support deep learning models for action forecasting, where the graph is used to encode a series of zones visited during a complex action (e.g. the user is first at the sink, and then move to the cutting board) to predict the next activity (e.g. cut something). Interestingly, this source of knowledge could be pivotal to enable a more natural and efficient learning of human skills with minimum motor impairment, opening interesting perspectives for imitation learning -based strategies to transfer human action skills to manipulators, or to inform a robot of the next human action, for contact avoidance and cooperation.

With a similar yet different purpose, graphs represent convenient mathematical structures to model human-object interactions. As an example, in [33] and [34] the authors propose a graph to encode contacts between hands and objects. The nodes state contains the information of which object is in contact with the two hands in the current and future time-spans, with the purpose of anticipating future actions of the user. A complex activity is then represented by a sequence of nodes, where edges represent temporal relationships (before/after) providing temporal context to the action recognition neural architecture.

In [131], the authors tackle the problem of visual imitation, meaning the ability of an agent to learn skills by observing and imitating a human demonstrator. Learning from complex and unstructured human behaviours requires a fine-grained understanding of the demonstrator's visual scene and of how it changes over time, which is addressed in [131] through the definition of hierarchical graph video representations, named Visual Entity Graphs (VEGs). The general idea is to encode human behaviour through a set of nodes representing visual entities - such as objects or hands - tracked in space and time, and connected through edges encoding their relative 3D spatial arrangement. Such structured representation enables a more fine-grained understanding of the activity, the key visual entities in the scene, and their spatial arrangement in the demonstrator's and imitator's environments. This encoding enables the agent to better understand the visual scene and changes over time, which is essential for successful visual imitation. The graph structure also allows for more efficient computation of visual similarity between the demonstrator's and imitator's environments, which is maximised through the imitator's actions to result in a correct transfer of skills.

Of relevance for this topic is also [132], where the authors address the challenges of estimating from videos the three-dimensional poses of hand and grasped objects in real-time during manipulation tasks. In this case, the graph encodes the spatial relationship between hand joints and object corners, and can dynamically adjust its structure depending on the input data, allowing the model to handle different hand-object configurations. Interestingly, the authors demonstrated that GNNs can make the difference in both refining the estimation of 2D keypoints, and also in converting them into their final 3D representation, also enabling on-line processing of visual streams.

## IV. GNNS FOR MULTI-AGENT SYSTEMS

In multi-agent systems, an effective coordination and communication between robots is of paramount importance to enable a fruitful cooperation, which is clearly critical for several downstream tasks. Traditionally, this problem has been addressed by exploiting centralised approaches, where all the intelligence and computation complexity is centralised

on a common controller node, which collects information from the environment, plan and then communicate single actions to each robot to fulfil the desired task. Such an approach has the strong benefit of having a central control system able to reason on the current (and past) status of the whole set of agents and the surrounding environment, thus requiring only little computation on the device side for action execution. However, this approach has a low fault tolerance: if even a single agent fails, all the actions of the robots need to be recomputed. Centralised solutions also badly scale with the increase of agents, and for difficult tasks it even requires additional per-robot computation [133].

To address these issues, recently many researchers devoted their attention to decentralised approaches, where each agent learns its set of actions according to environmental data and shared information from other agents. In this configuration, each robot has to deal with limited observability due to the range of the sensors, each robot can only see a partial portion of the environment and can communicate with the nearest agents. GNNs offer generalized and flexible structural representations of the elements in multi-robot systems, and have been recently widely used in decentralised approaches to adress several tasks.

The utilisation of GNNs can change according to the information that are encoded in the graph. Many notable works [43], [134], [135], [136], [137], [138], [139], [140], [141], [142], [143] focus the attention on the communication between robots, a crucial aspect for task and motion planning, and exploit graph encoding to efficiently model inter-agents relationships. Usually, graph neural networks are used to learn a communication graph, which is then exploited for example to directly support the agents in making decisions, or to foster multi-agent reinforcement learning frameworks in policy learning.

On the other hand, several methods propose to encode the information related to the environment topology, in the form of waypoints or point-of-interest [10], [11], [39], [144], [145], [146]. Works as [10], [11], and [39] focus on the spatio-temporal relationship between entities in the environment to build a graph, and use GNNs to learn environmental features for the robot. Other methods [144], [145], instead, propose a more comprehensive topological representation where both robots and environments are directly encoded in the graph. In particular, [144] is the first work that introduces a shared agent-entity graph in the context of multi-agent learning. Both environments entities and agents are encoded as nodes in the graph, and the edges reflect the communication links, which can be assumed to be always available (fully connected graph) or constrained by some sort of maximum distance range (i.e. edge exists only if the distance between nodes is below a given threshold). The communication inside the graph is performed in two steps: first, each agent computes its state embedding and aggregates the environment information with an attention mechanism [147], then, an inter-agent communication is performed to share local information along the team members.

Generally speaking, graphs provide several desirable advantages for multi-robot systems: i) invariance to the number of agents/points of interest ii) invariance to the permutation of the agents iii) suitability to learn policies for Multi Agent Reinforcement Learning (MARL) and imitation learning iv) policies learned on small systems are transferable to systems with a larger amount of agents. In the following, we will showcase how these benefits have been exploited for two relevant multi-robot scenarios: task and motion planning, where a fleet of robots move coordinately in an environment and cooperates to accomplish one or more activities; exploration and navigation, where agents are tasked to explore an unknown environment.

### A. TASK AND MOTION PLANNING

Graph structure is a natural way to describe decentralised logic. This implementation provides a way to plan multi-vehicle trajectories with performances similar to centralised approaches but in a decentralised manner, with a massive reduction of problem complexity and communication burden. In multi-robot path planning, the goal is to find collision-free paths to bring each agent in the team from a starting position to the target one.

The task can be particularly challenging when the agents have constraints in terms of range of observability and communication, quite common in real-world applications. Communication between agents is always crucial in multi-robot systems, and to fulfil this task it is particularly important to decide which information to share, when, and how, so that each agent is able to take effective decisions. To address this problem, [148] proposes to mix a convolutional neural network and a graph convolutional neural network to extract visual features from local observations of each agent, and to exchange such information within the team respectively. The graph neural network is structured as in [38], where each agent is a node and the edges are communication connections determined by a distance threshold. Such joint configuration helps the network to identify the relevant information to share with the team, in order to efficiently perform path planning. The work is later extended in [42], and an attention-like mechanism [24] is inserted in the graph neural network to model the intra-robot communication, where the weights on edges between nodes are proportional to the importance of the received message. A similar structure is exploited by [149], where an history of past paths is stored within the graph neural network to perform motion planning.

One challenge trend of research in multi-robot systems is the scalability to large numbers of agents, mostly because of the increasing communication burden. Graph neural networks offer a good solution to this issue, because they can mitigate the computation cost by aggregating information locally and not throughout the entire system. Interestingly,

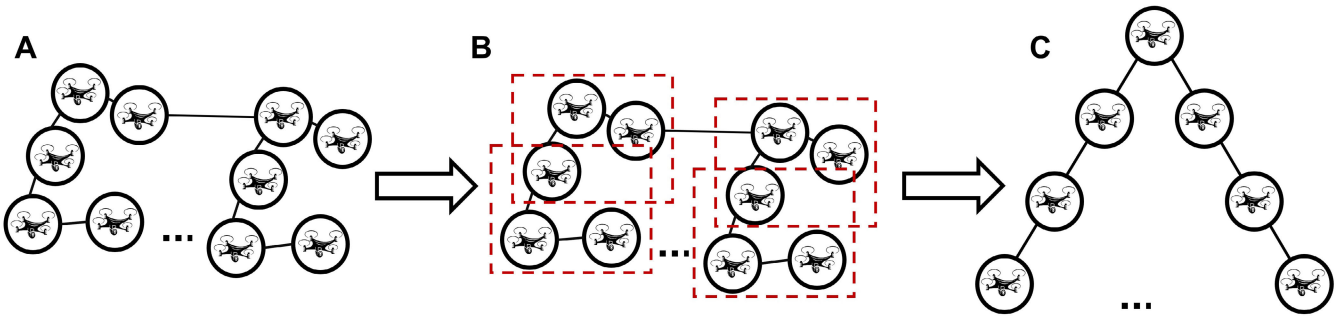



**FIGURE 8.** Example of a communication graph used to learn policies to produce a desired behaviour. Redrawn from [150].

a few works also demonstrated the feasibility to learn over a small number of agents a general model, which can be seamlessly extended to larger fleets [150], [151], [152]. Such works introduce Graph Policy Gradients, an algorithm that exploits graph neural networks to learn policies for the robots. A GNN is trained in the classic configuration for multi-robot systems, where each node is an agent and the edges connect team members that are within a threshold distance. Such a network can then be applied to several problems such as formation flying and path planning (see e.g. Fig. 8). The paper also discusses the scalability of GNNs in this framework toward an arbitrary number of robots, showing interesting perspectives in terms of limitation of computational cost and ability to transfer the learned policy to larger fleers in zero-shot, remarkably training on systems with cardinality from as little as three robots all the way up to a hundred.

A task that shares some similarities with path planning is to learn scheduling policies, which consists of coordinate agents programmed to complete tasks in predefined times and locations. To overcome similar issues of path planning, namely the difficulty to scale to large systems and lack of generality, GNNs have been investigated as a way to address them in RoboGNN [43], where the authors propose a graph attention -based model able to solve simple temporal network-based scheduling problems.

### B. EXPLORATION AND NAVIGATION

In multi-robot exploration, a team of robots needs to explore an unknown environment and additionally visit several regions of interest. In such a scenario, it is particularly important the communication and cooperation between agents to efficiently cover the exploration ground, and avoid conflicts or repeated explorations. Furthermore, in order to successfully fulfil the task, a key aspect is to define the spatial relationship between robots and regions of interest and extract environmental information. Graph neural networks provide a promising tool to efficiently regulate the communication between agents and aggregate topological information from the environment.

In section IV, we reviewed several approaches that build a communication graph for multi-robot systems coordination, which can be also applied for exploration tasks [43], [134], [135], [136], [137], [138], [139], [140], [141], [142]. Some of these works, instead, are specifically designed for the task of exploration, such as [142] where the authors tackle the task of coverage control, with the aim of predicting the distribution of a set of robots in a region such that the likelihood to spot events of interest is maximised. Of relevance is also [138], where the communication graph between several agents and a data sink is used to prevent data loss and allow robots to explore the environment efficiently in the presence of intermittent connectivity, for instance in space exploration.

It is also relevant to mention a line of research where graphs are used to model together both the environment and the agents therein. For the sake of completeness, we first introduce some relevant works which target this problem for single-robot exploration. In [10], [11], and [39] the authors propose an exploration graph to predict the next action [39], further extended with the addition of deep reinforcement learning to learn robot policies [10], [11]. The model provides a general representation of the SLAM-dependent robot state and environment, creating a spatio-temporal graph where robot state, landmarks, and frontiers are encoded as nodes, and a graph neural network is used to extract meaningful features from the environment. Topological graphs can be also used to model an entire environment, such as in [153], where the graph is used to model an indoor space for mobile robot exploration. The idea of processing environmental information is then later extended also to multi-robot exploration [12], [145], [146]. In [12], the authors propose to segment the exploration environment into the graph domain. To do this, the map is divided into regions and each of them is mapped into a node of the graph, while edges between nodes represent spatial distance. Based on the topological graph, a graph-based reinforcement learning algorithm is applied to learn how to assign exploration targets to each robot. Instead, in [145] and [146] the authors introduce the use of a spatial graph, where both map locations and agents are represented as nodes, and the connectivity encodes the set of allowed moves. In particular, [146] uses behaviour cloning to train a GNN controller to imitate the expert solution. Such a model can be easily generalised to scenarios with a larger map and

more agents. The usage of a GNN allows to easily choose the range in which the exchange of information may happen, by just varying the receptive field of the network.

In [145], Zhang et al. propose a coarse-to-fine exploration framework based on graph neural networks. Depending on the type of exploration, different regions of the environment may have different degree of relevance. As an example, for coarse exploration, the boundary is probably the most relevant zone where to focus, while for a fine inspection, looking to closer regions could be more informative. Graph neural networks can easily reflect such necessity giving more or less importance to features coming from different parts of the graph.

## V. STRENGTHS AND LIMITATIONS OF CURRENT LITERATURE

Moving from our analysis of the literature, we believe that it is relevant to mention how graph-based representations opened interesting research perspectives, such as for the modelling of functional, spatial, or temporal relationships between passive or active elements (between robots, objects, and zones of an environment). Interestingly, in the last few years, we observed an exponential increase in the adoption of this learning method for robotics applications, which at the moment is consolidated as one of the most prominent, together with images, for complex tasks.

For the field to progress to a higher maturity phase, however, we strongly believe that some aspects still deserve additional work. First, the analysis we performed strongly motivates and foster a more extensive and informed use of graphs as a convenient tool to represent unstructured information and enable learning on complex data structures. Yet, in many cases, graph learning is used in a naive fashion, albeit specific applications may benefit from advanced graph neural architectures formalisation such as attention based (see Fig. 3). One of the purposes of this survey is to bridge this gap by collecting and providing a cheat sheet on the most advanced graph learning theory for the robotics community.

## VI. CONCLUSION AND FUTURE PERSPECTIVES

The main goal of this paper is to showcase a comprehensive overview on the use of neural networks for graphs in robotics applications. Indeed, in the last decade, roboticists highly benefited from the advancement of techniques and methods defined in the machine learning community, serving as one of the preferred test benches for deep learning. However, often practical robotics use-cases deal with complex and unstructured data, such as three-dimensional data (point clouds), functional and temporal relationships between elements, etc. To unlock the potential of machine and deep learning for these applications, it is crucial to make models able to process unstructured data representation, for which graph encoding seems to be the preferable approach. We revised and categorised more than 100 papers on the topic, trying to identify the major trends of research in robotics in which graph encoding and learning play a crucial role in enabling novel tasks and making efficient the learning process.

Our literature review suggests that graph learning can enable a plethora of novel tasks, ranging from action recognition and forecasting, to space representation all the way to soft-bodies modelling. Interestingly, the number of novel applications is increasing with time, and we hope that our effort of collecting and revising the major contributions to the topic may provide a boost to the research on robotics-enabled novel and more complex tasks.

Finally, we also believe that the robotics community may not only serve as a test bench for machine learning, but could also contribute actively to the development of the field. Realistic problems coming from the robotics field could provide insights, constraints, and guidelines to foster novel learning and modelling scheme for researchers working on the foundation of graph learning. For this reason, this survey is also addressed to machine learning scientists, delivering a complete overview of the major challenges that the robotics community is currently addressing, with the hope of providing interesting research hints to bridge the gap between theory and application.

## REFERENCES

[1] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley, "Molecular graph convolutions: Moving beyond fingerprints," *J. Comput.-Aided Mol. Des.*, vol. 30, no. 8, pp. 595–608, Aug. 2016.

[2] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia, "Learning to simulate complex physics with graph networks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 8459–8468.

[3] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2016.

[4] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, May 2019, pp. 417–426.

[5] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 29–38.

[6] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Oct. 2019.

[7] D. Valsesia, G. Fracastoro, and E. Magli, "Deep graph-convolutional image denoising," *IEEE Trans. Image Process.*, vol. 29, pp. 8226–8237, 2020.

[8] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, "Hollywood in homes: Crowdsourcing data collection for activity understanding," in *Proc. ECCV*. Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 510–526.

[9] T. Nagarajan, Y. Li, C. Feichtenhofer, and K. Grauman, "Ego-Topo: Environment affordances from egocentric video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 160–169.

[10] F. Chen, J. D. Martin, Y. Huang, J. Wang, and B. Englot, "Autonomous exploration under uncertainty via deep reinforcement learning on graphs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 6140–6147.

[11] F. Chen, P. Szenher, Y. Huang, J. Wang, T. Shan, S. Bai, and B. Englot, "Zero-shot reinforcement learning on graphs for autonomous exploration under uncertainty," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 5193–5199.

[12] T. Luo, B. Subagdja, D. Wang, and A.-H. Tan, "Multi-agent collaborative exploration through graph-based deep reinforcement learning," in *Proc. IEEE Int. Conf. Agents (ICA)*, Oct. 2019, pp. 2–7.

[13] C. Packer, K. Gao, J. Kos, P. Krähenbühl, V. Koltun, and D. Song, "Assessing generalization in deep reinforcement learning," 2018, *arXiv:1810.12282*.
[14] D. Bacciu, F. Errica, A. Micheli, and M. Podda, "A gentle introduction to deep learning for graphs," *Neural Netw.*, vol. 129, pp. 203–221, Sep. 2020.
[15] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.
[16] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5782–5799, May 2023.
[17] J. Zhou, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020.
[18] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.
[19] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," 2015, *arXiv:1506.05163*.
[20] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.
[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
[22] F. Monti, K. Otness, and M. M. Bronstein, "MotifNet: A Motif-based graph convolutional network for directed graphs," in *Proc. IEEE Data Sci. Workshop (DSW)*, Jun. 2018, pp. 225–228.
[23] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.
[24] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Represent.*, 2017.
[25] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5425–5434.
[26] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
[27] N. Verma, E. Boyer, and J. Verbeek, "FeaStNet: Feature-steered graph convolutions for 3D shape analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2598–2606.
[28] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Int. Conf. Learn. Representations*, 2018.
[29] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10288–10297.
[30] G. Corso, L. Cavalleri, D. Beaini, P. Liò, and P. Veličković, "Principal neighbourhood aggregation for graph nets," 2020, *arXiv:2004.05718*.
[31] Y. Deng, C. Xia, X. Wang, and L. Chen, "Graph-transporter: A graph-based learning method for goal-conditioned deformable object rearranging task," in *Proc. IEEE Int. Conf. Syst. Man, Cybern. (SMC)*, Oct. 2022, pp. 1910–1916.
[32] A. Murali, W. Liu, K. Marino, S. Chernova, and A. Gupta, "Same object, different grasps: Data and semantic knowledge for task-oriented grasping," in *Proc. Conf. Robot Learn.*, 2020, pp. 1540–1557.
[33] E. Dessalene, M. Maynord, C. Devaraj, C. Fermuller, and Y. Aloimonos, "Egocentric object manipulation graphs," 2020, *arXiv:2006.03201*.
[34] E. Dessalene, C. Devaraj, M. Maynord, C. Fermüller, and Y. Aloimonos, "Forecasting action through contact representations from first person video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 6703–6714, Jun. 2023.
[35] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. W. Battaglia, "Learning mesh-based simulation with graph networks," 2020, *arXiv:2010.03409*.
[36] I. Huang, Y. Narang, R. Bajcsy, F. Ramos, T. Hermans, and D. Fox, "DefGraspNets: Grasp planning on 3D fields with graph neural nets," 2023, *arXiv:2303.16138*.
[37] T. Wang, R. Liao, J. Ba, and S. Fidler, "NerveNet: Learning structured policy with graph neural networks," in *Proc. Int. Conf. Learn. Represent.*, vol. 30. Vancouver, BC, Canada, 2018.
[38] A. Prorok, "Graph neural networks for learning robot team coordination," 2018, *arXiv:1805.03737*.
[39] F. Chen, J. Wang, T. Shan, and B. Englot, "Autonomous exploration under uncertainty via graph convolutional networks," in *Proc. Int. Symp. Robot. Res.*, 2019, pp. 676–691.
[40] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," 2015, *arXiv:1511.05493*.
[41] M. Ding, Y. Liu, C. Yang, and X. Lan, "Visual manipulation relationship detection based on gated graph neural network for robotic grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 1404–1410.
[42] Q. Li, W. Lin, Z. Liu, and A. Prorok, "Message-aware graph attention networks for large-scale multi-robot path planning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5533–5540, Jul. 2021.
[43] Z. Wang and M. Gombolay, "Learning scheduling policies for multi-robot coordination with graph attention networks," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4509–4516, Jul. 2020.
[44] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 2224–2232.
[45] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, "Quantum-chemical insights from deep tensor neural networks," *Nature Commun.*, vol. 8, no. 1, Jan. 2017, Art. no. 13890.
[46] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," 2014, *arXiv:1406.1078*.
[47] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," 2018, *arXiv:1810.01566*.
[48] X. Ma, D. Hsu, and W. S. Lee, "Learning latent graph dynamics for visual manipulation of deformable objects," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 8266–8273.
[49] C. Wang, Y. Zhang, X. Zhang, Z. Wu, X. Zhu, S. Jin, T. Tang, and M. Tomizuka, "Offline-online learning of deformation model for cable manipulation with graph neural networks," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 5544–5551, Apr. 2022.
[50] H. Shi, H. Xu, Z. Huang, Y. Li, and J. Wu, "RoboCraft: Learning to see, simulate, and shape elasto-plastic objects with graph networks," 2022, *arXiv:2205.02909*.
[51] Y. Deng, X. Wang, and L. Chen, "Learning visual-based deformable object rearrangement with local graph neural networks," *Complex Intell. Syst.*, vol. 9, pp. 5923–5936, Apr. 2023.
[52] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 1–12, Jul. 2014.
[53] P. Battaglia, R. Pascanu, M. Lai, and D. Jimenez Rezende, "Interaction networks for learning about objects, relations and physics," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 4509–4517.
[54] Y. Li, J. Wu, J.-Y. Zhu, J. B. Tenenbaum, A. Torralba, and R. Tedrake, "Propagation networks for model-based control under partial observation," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 1205–1211.
[55] K. Holkar and L. Waghmare, "An overview of model predictive control," *Int. J. Control Autom.*, vol. 3, no. 4, pp. 47–63, 2010.
[56] X. Lin, Y. Wang, Z. Huang, and D. Held, "Learning visible connectivity dynamics for cloth smoothing," in *Proc. Conf. Robot Learn.*, 2022, pp. 256–266.
[57] K. Mo, Y. Deng, C. Xia, and X. Wang, "Learning language-conditioned deformable object manipulation with graph dynamics," 2023, *arXiv:2303.01310*.
[58] J. D. Almeida, P. Schydlo, A. Dehban, and J. Santos-Victor, "SENSORI-MOTOR GRAPH: Action-conditioned graph neural network for learning robotic soft hand dynamics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 9569–9576.

[59] M. Oliva, S. Banik, J. Josifovski, and A. Knoll, "Graph neural networks for relational inductive bias in vision-based deep reinforcement learning of robot control," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2022, pp. 1–9.

[60] C. D. Santina, C. Duriez, and D. Rus, "Model-based control of soft robots: A survey of the state of the art and open challenges," *IEEE Control Syst.*, vol. 43, no. 3, pp. 30–65, Jun. 2023.

[61] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer Handbook of Robotics*. Berlin, Germany: Springer, 2016, pp. 955–988.

[62] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proc. ICRA Millennium Conf. IEEE Int. Conf. Robot. Automat. Symposia*, Apr. 2000, pp. 348–353.

[63] K. Kleeberger, R. Bormann, W. Kraus, and M. F. Huber, "A survey on learning-based robotic grasping," *Current Robot. Rep.*, vol. 1, no. 4, pp. 239–249, Dec. 2020.

[64] B. Wen, W. Lian, K. Bekris, and S. Schaal, "CaTGrasp: Learning category-level task-relevant grasping in clutter from simulation," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 6401–6408.

[65] L. Berscheid, P. Meißner, and T. Kröger, "Robot learning of shifting objects for grasping in cluttered environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 612–618.

[66] M. R. Dogar, K. Hsiao, M. Ciocarlie, and S. Srinivasa, *Physics-Based Grasp Planning Through Clutter*. Cambridge, MA, USA: MIT Press, 2012.

[67] D. Berenson, "Manipulation of deformable objects without modeling and simulating deformation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 4525–4532.

[68] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Sci. Robot.*, vol. 6, no. 54, May 2021, Art. no. eabd8803.

[69] L. Zaidi, J. A. Corrales, B. C. Bouzgarrou, Y. Mezouar, and L. Sabourin, "Model-based strategy for grasping 3D deformable objects using a multi-fingered robotic hand," *Robot. Auto. Syst.*, vol. 95, pp. 196–206, Sep. 2017.

[70] E. Corona, G. Alenyà, A. Gabas, and C. Torras, "Active garment recognition and target grasping point detection using deep learning," *Pattern Recognit.*, vol. 74, pp. 629–641, Feb. 2018.

[71] R. Jangir, G. Alenyà, and C. Torras, "Dynamic cloth manipulation with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 4630–4636.

[72] X. Li, G. Chi, S. Vidas, and C. C. Cheah, "Human-guided robotic comanipulation: Two illustrative scenarios," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 5, pp. 1751–1763, Sep. 2016.

[73] L. Peternel, N. Tsagarakis, D. Caldwell, and A. Ajoudani, "Adaptation of robot physical behaviour to human fatigue in human-robot co-manipulation," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots (Humanoids)*, Nov. 2016, pp. 489–494.

[74] F. Dimeas and N. Aspragathos, "Reinforcement learning of variable admittance control for human-robot co-manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 1011–1016.

[75] M. L. Elwin, B. Strong, R. A. Freeman, and K. M. Lynch, "Human-multirobot collaborative mobile manipulation: The omnid mocobots," *IEEE Robot. Autom. Lett.*, vol. 8, no. 1, pp. 376–383, Jan. 2023.

[76] F. Xie, A. Chowdhury, M. De Paolis Kaluza, L. Zhao, L. Wong, and R. Yu, "Deep imitation learning for bimanual robotic manipulation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 2327–2337.

[77] Y. Lin, A. S. Wang, E. Undersander, and A. Rai, "Efficient and interpretable robot manipulation with graph neural networks," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2740–2747, Apr. 2022.

[78] J. H. Kwak, J. Lee, J. J. Whang, and S. Jo, "Semantic grasping via a knowledge graph of robotic manipulation: A graph representation learning approach," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9397–9404, Oct. 2022.

[79] L. Collodi, D. Bacciu, M. Bianchi, and G. Averta, "Learning with few examples the semantic description of novel human-inspired grasp strategies from RGB data," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2573–2580, Apr. 2022.

[80] M. Neumann, P. Moreno, L. Antanas, R. Garnett, and K. Kersting, "Graph kernels for object category prediction in task-dependent robot grasping," in *Proc. 11th Workshop Mining Learn. With Graphs*, 2013, pp. 1–6.

[81] A. Alliegro, M. Rudorfer, F. Frattin, A. Leonardis, and T. Tommasi, "End-to-end learning to grasp via sampling from object point clouds," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9865–9872, Oct. 2022.

[82] A. Iriondo, E. Lazkano, and A. Ansuategi, "Affordance-based grasping point detection using graph convolutional networks for industrial bin-picking applications," *Sensors*, vol. 21, no. 3, p. 816, Jan. 2021.

[83] A. Garcia-Garcia, B. S. Zapata-Impata, S. Orts-Escolano, P. Gil, and J. Garcia-Rodriguez, "TactileGCN: A graph convolutional network for predicting grasp stability with tactile sensors," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–8.

[84] G. Zuo, J. Tong, Z. Wang, and D. Gong, "A graph-based deep reinforcement learning approach to grasping fully occluded objects," *Cognit. Comput.*, vol. 15, no. 1, pp. 36–49, Jan. 2023.

[85] X. Lou, Y. Yang, and C. Choi, "Learning object relations with graph neural networks for target-driven grasping in dense clutter," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 742–748.

[86] Y. Huang, A. Conkey, and T. Hermans, "Planning for multi-object manipulation with graph neural network relational classifiers," 2022, *arXiv:2209.11943*.

[87] R. Li, A. Jabri, T. Darrell, and P. Agrawal, "Towards practical multi-object manipulation using relational reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 4051–4058.

[88] I. Kapelyukh and E. Johns, "My house, my rules: Learning tidying preferences with graph neural networks," in *Proc. Conf. Robot Learn.*, 2022, pp. 740–749.

[89] M. Wilson and T. Hermans, "Learning to manipulate object collections using grounded state representations," in *Proc. Conf. Robot Learn.*, 2020, pp. 490–502.

[90] T. Silver, R. Chitnis, A. Curtis, J. B. Tenenbaum, T. Lozano-Pérez, and L. P. Kaelbling, "Planning with learned object importance in large problem instances using graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 13, 2021, pp. 11962–11971.

[91] A. Mousavian, C. Eppner, and D. Fox, "6-DOF GraspNet: Variational grasp generation for object manipulation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2901–2910.

[92] H. Liang, X. Lou, Y. Yang, and C. Choi, "Learning visual affordances with target-orientated deep Q-network to grasp objects by harnessing environmental fixtures," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 2562–2568.

[93] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, "6-DOF grasping for target-driven object manipulation in clutter," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6232–6238.

[94] X. Lou, Y. Yang, and C. Choi, "Collision-aware target-driven object grasping in constrained environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 6364–6370.

[95] Y. Liu, R. Bao, J. Tao, J. Li, M. Dong, and C. Pan, "Recent progress in tactile sensors and their applications in intelligent systems," *Sci. Bull.*, vol. 65, no. 1, pp. 70–88, Jan. 2020.

[96] A. Yamaguchi and C. G. Atkeson, "Recent progress in tactile sensing and sensors for robotic manipulation: Can we turn tactile sensing into vision?" *Adv. Robot.*, vol. 33, no. 14, pp. 661–673, Jul. 2019.

[97] Z. Xia, Z. Deng, B. Fang, Y. Yang, and F. Sun, "A review on sensory perception for dexterous robotic manipulation," *Int. J. Adv. Robotic Syst.*, vol. 19, no. 2, Mar. 2022, Art. no. 172988062210959.

[98] N. F. Lepora, "Soft biomimetic optical tactile sensing with the TacTip: A review," *IEEE Sensors J.*, vol. 21, no. 19, pp. 21131–21143, Oct. 2021.

[99] J. Xu, S. Song, and M. Ciocarlie, "TANDEM: Learning joint exploration and decision making with tactile sensors," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10391–10398, Oct. 2022.

[100] S. Funabashi, T. Isobe, F. Hongyi, A. Hiramoto, A. Schmitz, S. Sugano, and T. Ogata, "Multi-fingered in-hand manipulation with various object properties using graph convolutional networks and distributed tactile sensors," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2102–2109, Apr. 2022.

[101] M. Tenorth and M. Beetz, "Representations for robot knowledge in the KnowRob framework," *Artif. Intell.*, vol. 247, pp. 151–169, Jun. 2017.

[102] A. Saxena, A. Jain, O. Sener, A. Jami, D. K. Misra, and H. S. Koppula, "RoboBrain: Large-scale knowledge engine for robots," 2014, *arXiv:1412.0691*.

[103] A. Daruna, W. Liu, Z. Kira, and S. Chetnova, "RoboCSE: Robot common sense embedding," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 9777–9783.

[104] A. Daruna, L. Nair, W. Liu, and S. Chernova, "Towards robust one-shot task execution using knowledge graph embeddings," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11118–11124.

[105] P. Ardón, È. Pairet, R. P. A. Petrick, S. Ramamoorthy, and K. S. Lohan, "Learning grasp affordance reasoning through semantic relations," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4571–4578, Oct. 2019.

[106] G. A. Miller, "WordNet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[107] Y. Kong and Y. Fu, "Human action recognition and prediction: A survey," *Int. J. Comput. Vis.*, vol. 130, no. 5, pp. 1366–1401, May 2022.

[108] Y. Wang, S. Cang, and H. Yu, "A survey on wearable sensor modality centred human activity recognition in health care," *Expert Syst. Appl.*, vol. 137, pp. 167–190, Dec. 2019.

[109] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auto. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

[110] S. Ekvall and D. Kragic, "Robot learning from demonstration: A task-level planning approach," *Int. J. Adv. Robotic Syst.*, vol. 5, no. 3, p. 33, Sep. 2008.

[111] B. Fang, S. Jia, D. Guo, M. Xu, S. Wen, and F. Sun, "Survey of imitation learning for robotic manipulation," *Int. J. Intell. Robot. Appl.*, vol. 3, no. 4, pp. 362–369, Dec. 2019.

[112] Y. Tsurumine and T. Matsubara, "Goal-aware generative adversarial imitation learning from imperfect demonstration for robotic cloth manipulation," *Robot. Auto. Syst.*, vol. 158, Dec. 2022, Art. no. 104264.

[113] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar, "MimicPlay: Long-horizon imitation learning by watching human play," 2023, *arXiv:2302.12422*.

[114] A. Albini, S. Denei, and G. Cannata, "Human hand recognition from robotic skin measurements in human-robot physical interactions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 4348–4353.

[115] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, "An atlas of physical human–robot interaction," *Mech. Mach. Theory*, vol. 43, no. 3, pp. 253–270, Mar. 2008.

[116] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 961–971.

[117] Y. Huang, H. Bi, Z. Li, T. Mao, and Z. Wang, "STGAT: Modeling spatial–temporal interactions for human trajectory prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6271–6280.

[118] A. Thobbi, Y. Gu, and W. Sheng, "Using human motion estimation for human-robot cooperative manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 2873–2878.

[119] M. Natarajan and M. Gombolay, "Effects of anthropomorphism and accountability on trust in human robot interaction," in *Proc. 15th ACM/IEEE Int. Conf. Hum.-Robot Interact. (HRI)*, Mar. 2020, pp. 33–42.

[120] J. Wang, X. Long, Y. Gao, E. Ding, and S. Wen, "Graph-PCNN: Two stage human pose estimation with graph pose refinement," in *Proc. Eur. Conf. Comput. Vis.* Glasgow, U.K.: Springer, Aug. 2020, pp. 492–508.

[121] S. Jin, W. Liu, E. Xie, W. Wang, C. Qian, W. Ouyang, and P. Luo, "Differentiable hierarchical graph grouping for multi-person pose estimation," in *Proc. Eur. Conf. Comput. Vis.* Glasgow, U.K.: Springer, Aug. 2020, pp. 718–734.

[122] Z. Qiu, K. Qiu, J. Fu, and D. Fu, "DGCN: Dynamic graph convolutional network for efficient multi-person pose estimation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, Apr. 2020, pp. 11924–11931.

[123] W. Peng, X. Hong, H. Chen, and G. Zhao, "Learning graph convolutional network for skeleton-based human action recognition by neural searching," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 2669–2676.

[124] J. Wang, S. Tan, X. Zhen, S. Xu, F. Zheng, Z. He, and L. Shao, "Deep 3D human pose estimation: A review," *Comput. Vis. Image Understand.*, vol. 210, Sep. 2021, Art. no. 103225.

[125] T. Ahmad, L. Jin, X. Zhang, S. Lai, G. Tang, and L. Lin, "Graph convolutional neural network for human action recognition: A comprehensive survey," *IEEE Trans. Artif. Intell.*, vol. 2, no. 2, pp. 128–145, Apr. 2021.

[126] A. Núñez-Marcos, G. Azkune, and I. Arganda-Carreras, "Egocentric vision-based action recognition: A survey," *Neurocomputing*, vol. 472, pp. 175–197, Feb. 2022.

[127] J. Carreira and A. Zisserman, "Quo Vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4724–4733.

[128] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, and W. Price, "Scaling egocentric vision: The EPIC-KITCHENS dataset," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 720–736.

[129] K. Grauman, "Ego4D: Around the world in 3,000 hours of egocentric video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 18973–18990.

[130] F. Sener, D. Chatterjee, D. Shelepov, K. He, D. Singhania, R. Wang, and A. Yao, "Assembly101: A large-scale multi-view video dataset for understanding procedural activities," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 21064–21074.

[131] M. Sieb, Z. Xian, A. Huang, O. Kroemer, and K. Fragkiadaki, "Graph-structured visual imitation," in *Proc. Conf. Robot Learn.*, 2020, pp. 979–989.

[132] B. Doosti, S. Naha, M. Mirbagheri, and D. J. Crandall, "HOPE-Net: A graph-based model for hand-object pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6607–6616.

[133] A. Khan, A. Ribeiro, V. Kumar, and A. G. Francis, "Graph neural networks for motion planning," 2020, *arXiv:2006.06248*.

[134] C. Sun, M. Shen, and J. P. How, "Scaling up multiagent reinforcement learning for robotic systems: Learn an adaptive sparse communication graph," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 11755–11762.

[135] E. Tolstaya, F. Gama, J. Paulos, G. Pappas, V. Kumar, and A. Ribeiro, "Learning decentralized controllers for robot swarms with graph neural networks," in *Proc. Conf. robot Learn.*, 2020, pp. 671–682.

[136] J. Blumenkamp and A. Prorok, "The emergence of adversarial communication in multi-agent reinforcement learning," in *Proc. Conf. Robot Learn.*, vol. 155, J. Kober, F. Ramos, and C. Tomlin, Eds., Nov. 2021, pp. 1394–1414.

[137] R. Kortvelesy and A. Prorok, "ModGNN: Expert policy approximation in multi-agent systems with a modular graph neural network architecture," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 9161–9167.

[138] L. Clark, J. Galante, B. Krishnamachari, and K. Psounis, "A queue-stabilizing framework for networked multi-robot exploration," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2091–2098, Apr. 2021.

[139] Z. Wang, C. Liu, and M. Gombolay, "Heterogeneous graph attention networks for scalable multi-robot scheduling with temporospatial constraints," *Auto. Robots*, vol. 46, pp. 249–268, Aug. 2021.

[140] E. Sebastian, T. Duong, N. Atanasov, E. Montijano, and C. Sagues, "LEMURS: Learning distributed multi-robot interactions," 2022, *arXiv:2209.09702*.

[141] F. Gama, Q. Li, E. Tolstaya, A. Prorok, and A. Ribeiro, "Synthesizing decentralized controllers with graph neural networks and imitation learning," *IEEE Trans. Signal Process.*, vol. 70, pp. 1932–1946, 2022.

[142] W. Gosrich, S. Mayya, R. Li, J. Paulos, M. Yim, A. Ribeiro, and V. Kumar, "Coverage control in multi-robot systems via graph neural networks," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 8787–8793.

[143] M. Tzes, N. Bousias, E. Chatzipantazis, and G. J. Pappas, "Graph neural networks for multi-robot active information acquisition," 2022, *arXiv:2209.12091*.

[144] A. Agarwal, S. Kumar, K. Sycara, and M. Lewis, "Learning transferable cooperative behavior in multi-agent teams," in *Proc. 19th Int. Conf. Auto. Agents MultiAgent Syst.* Richland, SC, USA: International Foundation for Autonomous Agents and Multiagent Systems, 2020, pp. 1741–1743.

[145] H. Zhang, J. Cheng, L. Zhang, Y. Li, and W. Zhang, "H2GNN: Hierarchical-hops graph neural networks for multi-robot exploration in unknown environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3435–3442, Apr. 2022.

[146] E. Tolstaya, J. Paulos, V. Kumar, and A. Ribeiro, "Multi-robot coverage and exploration using spatial graph neural networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 8944–8950.

[147] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6000–6010.

[148] Q. Li, F. Gama, A. Ribeiro, and A. Prorok, "Graph neural networks for decentralized multi-robot path planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 11785–11792.

[149] J. Li, Z. Su, and Y. Qiu, "Dynamic motion planning model for multirobot using graph neural network and historical information," *Adv. Intell. Syst.*, vol. 5, no. 8, Aug. 2023, Art. no. 2300036.

[150] A. Khan, E. Tolstaya, A. Ribeiro, and V. Kumar, "Graph policy gradients for large scale robot control," in *Proc. Conf. robot Learn.*, 2020, pp. 823–834.

[151] A. Khan, V. Kumar, and A. Ribeiro, "Graph policy gradients for large scale unlabeled motion planning with constraints," 2019, *arXiv:1909.10704*.

[152] A. Khan, V. Kumar, and A. Ribeiro, "Large scale distributed collaborative unlabeled motion planning with graph policy gradients," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5340–5347, Jul. 2021.

[153] C. Gomez, A. C. Hernandez, and R. Barber, "Topological frontier-based exploration and map-building using semantic information," *Sensors*, vol. 19, no. 20, p. 4595, Oct. 2019.

[154] N. Funk et al., "Learn2Assemble with structured representations and search for robotic architectural construction," in *Proc. Conf. Robot Learn.* PMLR, 2022, pp. 1401–1411.

[155] N. Funk et al., "Graph-based reinforcement learning meets mixed integer programs: An application to 3D robot assembly discovery," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. IEEE, 2022, pp. 10215–10222.

**FRANCESCA PISTILLI** (Member, IEEE) received the M.Sc. degree in electronic engineering from the Polytechnic of Turin, in 2019, the M.Sc. degree in electrical and computer engineering from the University of Illinois at Chicago, Chicago, IL, USA, in 2020, and the Ph.D. degree from the Image Processing and Learning Group (IPL), Polytechnic of Turin. She is currently a Postdoctoral Researcher of computer vision with the Polytechnic of Turin. Her current research interests include graph-convolutional neural networks, implicit neural representations, transformers, and other cutting-edge deep learning algorithms applied to images, point cloud processing, and robotics.

**GIUSEPPE AVERTA** (Member, IEEE) received the B.S. degree in biomedical engineering and the M.S. degree (Hons.) in robotics from the University of Pisa, in 2013 and 2016, respectively. He was a Postdoctoral Researcher with the University of Pisa. In 2019, he was a Visiting Student with the Eric P. and Evelyn E. Newman Laboratory, Biomechanics and Human Rehabilitation Group, MIT. He is currently an Assistant Professor of robotics and machine learning with the Polytechnic of Turin. He is also an Italian Institute of Technology Alumnus. His current research interests include the development of a truly embodied intelligence for human-robot cooperation, with research activities in the optimization of machine learning models for edge computing, deep learning for egocentric vision, human-inspired design, planning, and control guidelines for autonomous, collaborative, assistive, and prosthetic robots.

Open Access funding provided by 'Politecnico di Torino' within the CRUI CARE Agreement