

# ABNORMALITY DETECTION USING GRAPH MATCHING FOR MULTI-TASK DYNAMICS OF AUTONOMOUS SYSTEMS

Hassan Zaal<sup>1,2</sup>, Mohamad Baydoun<sup>1</sup>, Lucio Marcenaro<sup>1</sup>, Laurissa Tokarchuk<sup>2</sup>, and Carlo S. Regazzoni<sup>1</sup>

<sup>1</sup>Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture (DITEN), University of Genova, Italy

<sup>2</sup>School of Electronic Engineering and Computer Science, Queen Mary University of London, UK

{hassan.zaal, mohamad.baydoun}@ginevra.dibe.unige.it,

{carlo.regazzoni, lucio.marcenaro}@unige.it, laurissa.tokarchuk@qmul.ac.uk,

## Abstract

*Self-learning abilities in autonomous systems are essential to improve their situational awareness and detection of normal/abnormal situations. In this work, we propose a graph matching technique for activity detection in autonomous agents by using the Gromov-Wasserstein framework. A clustering approach is used to discretise continuous agents' states related to a specific task into a set of nodes with similar objectives. Additionally, a probabilistic transition matrix between nodes is used as edges weights to build a graph. In this paper, we extract an abnormal area based on a sub-graph that encodes the differences between coupled of activities. Such sub-graph is obtained by applying a threshold on the optimal transport matrix, which is obtained through the graph matching procedure. The obtained results are evaluated through experiments performed by a robot in a simulated environment and by a real autonomous vehicle moving within a University Campus.*

## 1. INTRODUCTION

The autonomous system refers to a self-capable system that can perform tasks by itself in its environment. To enable such autonomy, the system should be able to observe the surrounding environment and its states to perform suitable actions [1]. As the system could face many unseen experiences, it should be able to learn incrementally in an unsupervised manner. The concept of incremental learning refers to learning newly acquired knowledge without forgetting the previous knowledge [2, 3]. In autonomous systems, abnormality detection can be defined as the difference between expected and currently observed state changes in a

given region of the state space [4]. Detecting abnormality could be useful for incremental learning and transfer knowledge, where abnormality represents a “new knowledge” such as pedestrian avoidance. The adaptation of new situations (abnormalities) without losing the previous one allows the system to learn incrementally. In this paper, we extract these abnormalities.

It is useful to group data in abstract way and to represent them in a generic way such as graphs [5, 6], due to the following facts [7]:

- (i) Abstract state encodes similar states, which reduce the size and complexity.
- (ii) Graphs afford a powerful representation for nodes' interactions.

Given two or more graphs, it can be useful to find the correspondences between them, which is called a graph matching. Finding the correspondences between different graphs are increasingly set to become a vital factor in many applications such as in ([8, 9, 10]). To build useful micro-skills which are new observations with the associated micro-actions, it needs to match data from different sensors and control values. Such matching can be useful to determine the causality between observations and actions, which enables the interaction with the surrounding environment. Different sensors' data and actuators' values have different space dimensions. Hence, domain adaptation arises as a challenging issue, because it considers such cases where the data from different space dimensions. Domain adaptation estimates the unknown labels from the target graph using the label information on the source graph and the similarity between the two graphs [11]. Many methods are proposed for domain adaptation. Recently, Gromov-Wasserstein distance [12] is increasingly becoming a remarkable distance in machine learning community. This is due to the fact that

it measures the distance between samples in each domain, then it compares these distances, instead of comparing the sample from different domains and/or dimensions immediately.

The main contributions of this article are listed as follows:

- (i) Graph matching between different maneuvering tasks.
- (ii) Graph matching of different sensors' descriptors, which could enable us to determine the causality.
- (iii) Extracting sub-graph that is associated to the abnormal area.

## 2. RELATED WORK

Graph matching (GM) problem can be divided into two categories [13]: in the first category, which was the earlier works of graph matching, it has considered the exact matching, while the second category, it accepts to have distortions i.e., inexact matching. The inexact matching allows to use graph matching for many real world applications as it is more flexible.

In [14], the authors proposed a method to parameterize and learn a structural attributes of a graph model and optimize them to increase the matching accuracy. They proposed a histogram-attributed relational graph (HARG), where histogram distributions represent all node and edge attributes. In this method the learned graphs are dense not sparse, where each node is connected to all other nodes in the graph. While in our method it is not required to have a dense graph.

Abnormality detection problem has been investigated in different algorithms. In [15], a Markov Jump Particle Filter (MJPF) is used for abnormality detection, where it detects deviations from the learned model based on internal innovation measurements. In [16, 17], Generative Adversarial Nets (GANs) are used for abnormalities detection. While in our work, the purpose of abnormality detection using graph matching is to extract the abnormal areas and build a dictionary of micro-skills to use them for incremental learning and transfer knowledge.

## 3. METHOD

The proposed method is summarized in the block diagram presented in Fig. 1. Following sections are dedicated to explain each step of the proposed method.

### 3.1. Generalized state space

Let us denote the measurements from the sensor ( $m$ ) of an agent while doing a specific task ( $d$ ) at a time ( $k$ ) as  $Z_k^{m,d}$ , and the associated state to these measurements as  $X_k^{m,d}$  such that

$$Z_k^{m,d} = f(X_k^{m,d}) + w_k \quad (1)$$

where  $f(\cdot)$  is a mapping function between the states and observations and  $w_k$  denotes the noise from the sensor. We define the generalized state as:

$$\mathbf{X}_k^{m,d} = [X_k^{m,d} \ \dot{X}_k^{m,d} \ \ddot{X}_k^{m,d} \ \dots \ X_k^{(L)m,d}]^\top \quad (2)$$

where  $(L)$  are the  $L$ -th time derivative of the vector state.

### 3.2. Descriptors generation

Discrete descriptors are generated by grouping generalized states into a set of nodes (regions). Each region encodes the dynamics of generalized states that share a similar objective. A clustering approach called growing neural gas (GNG) [18] is used to group these generalized states. For each sensor/task, a GNG is trained. Each GNG receives  $\mathbf{X}_k^{m,d}$  and produces a set of learned nodes  $\mathbf{V}^{m,d}$  such that

$$\mathbf{V}^{m,d} = \{V_1^{m,d}, V_2^{m,d}, \dots, V_{N^{m,d}}^{m,d}\}, \quad (3)$$

where  $N^{m,d}$  is the number of nodes in the GNG associated to the module  $m$  of agent's task  $d$ .

### 3.3. Learning Transition matrix and Nodes properties

By observing the activated nodes over time, it is possible to estimate a transition matrix  $C_g^{m,d}$  which encodes a set of probabilities of passing from a discrete descriptor to another one. Where,  $g$  denotes to graph 1 (source  $g = s$ ) or graph 2 (target  $g = t$ ). Also, we obtained  $\mu_g^{m,d}$  which encodes the mean value of each node such as

$$\mu_g^{m,d} = \{\mu_1^{m,d}, \mu_2^{m,d}, \dots, \mu_{N^{m,d}}^{m,d}\}, \quad (4)$$

### 3.4. Graphs creation

Let  $G_s(\mathbf{V}_s^{m,d}, \epsilon_s^{m,d})$  and  $G_t(\mathbf{V}_t^{m,d}, \epsilon_t^{m,d})$  are the source and target graphs. The edges  $\epsilon_g^{m,d}$  in each graph are weighted based-on the measured interactions between its vertices. For  $g = s, t$ ,  $\epsilon_g^{m,d} = \{(v_i, v_j, w_{ij}) | (v_i, v_j) \in \mathbf{V}_g^{m,d}\}$ , where,  $w_{ij}$  is the weight of the edge between  $(v_i, v_j)$ .

We define metric-measure spaces correspond to  $(C_s^{m,d}, \mu_s^{m,d}), (C_t^{m,d}, \mu_t^{m,d})$ , where  $C_g^{m,d}$  represents a transition matrix obtained based-on the interactions and  $\mu_g^{m,d}$  represents a statistical information about each node.

### 3.5. Gromov-Wasserstein discrepancy

Inspired by the method in [19], a Gromov-Wasserstein framework is used to match graphs. The Gromov-Wasserstein discrepancy between  $(C_s, \mu_s)$  and  $(C_t, \mu_t)$  (for simplicity, we will not mention in this section about which sensor is used and which task is performed) is defined as in (5)

$$d_{GW}(\mu_s, \mu_t) := \min_{T \in \Pi(\mu_s, \mu_t)} \langle L(C_s, C_t, T), T \rangle \quad (5)$$

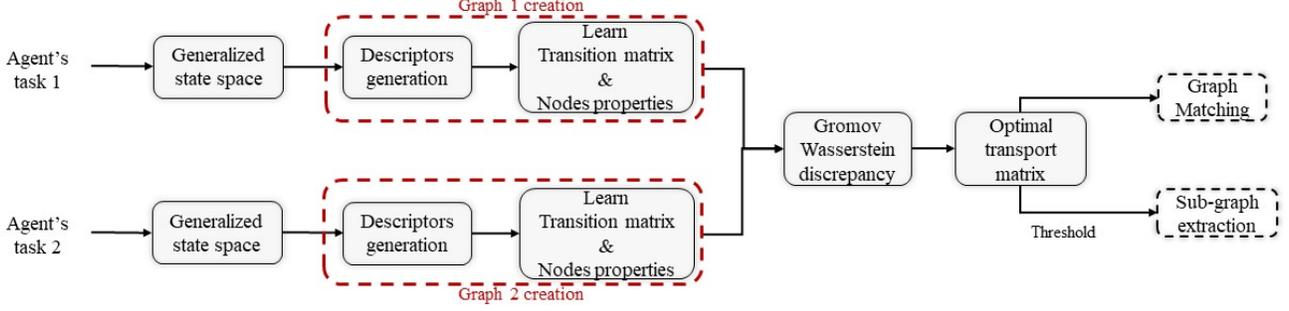


Figure 1. Block diagram of proposed method for the case of two tasks.

where,  $\langle \cdot, \cdot \rangle$  represents the inner products between two matrices, and  $L(\cdot, \cdot)$  is an element-wise loss function.

Taking into account the advantages of Gromov-Wasserstein distance that mentioned before, the optimization problem in (6) represents the method is used to find the optimal transport matrix between correspondences from two different graph and jointly learning latent vectors of graphs' nodes. It is performed by the first and second terms of (6), respectively, where  $\alpha$  is a parameter determines the influence of the latent vectors in this problem. The matrix  $D(E_s, E_t)$  measures the distances between node embeddings.

$$\min_{E_s, E_t} \min_{T \in \Pi(\mu_s, \mu_t)} \langle L(C_s(E_s), C_t(E_t), T), T \rangle + \alpha \langle D(E_s, E_t), T \rangle \quad (6)$$

where,  $\Pi(\mu_s, \mu_t) = \{T \in \mathbb{R}^{|V_s| \times |V_t|} \mid T \mathbf{1}_{|V_t|} = \mu_s, T^T \mathbf{1}_{|V_s|} = \mu_t\}$  and  $C_g(E_g) = (1-\alpha)E_g + \alpha D(E_g, E_g)$ , for  $g = s, t$ . Here, the first term in (6) represents Gromov-Wasserstein discrepancy between the source and target graph, while the second one is Wasserstein discrepancy for nodes' embeddings.

### 3.5.1 Learning correspondences

We solve the optimization problem in (6), by dividing it to two sub-problems and solving them alternatively. In this case, learning the optimal transport matrix based-on the obtained embeddings from each previous step (m) as in (7).

$$\min_{T \in \Pi(\mu_s, \mu_t)} \langle L(C_s(E_s^{(m)}), C_t(E_t^{(m)}), T), T \rangle + \alpha \langle D(E_s^{(m)}, E_t^{(m)}), T \rangle \quad (7)$$

Because of the first term in (7) is a non-convex quadratic term, we use a proximal point method (such as in [20]) to solve it. We add a regularize to (7), which will be the proximal term as following:

$$\min_{T \in \Pi(\mu_s, \mu_t)} \langle L(C_s(E_s^{(m)}), C_t(E_t^{(m)}), T), T \rangle + \alpha \langle D(E_s^{(m)}, E_t^{(m)}), T \rangle + \gamma KL(T || T^{(n)}) \quad (8)$$

Where KL is Kullback-Leibler divergence.

We solve this optimization problem by using Sinkhorn-Knopp algorithm[21]. This algorithm alternately normalizes the rows and the columns of the optimal transport matrix.

### 3.5.2 Learning latent vectors

Learning latent vectors for graph nodes aims to find the similarity between nodes in latent space. Learning such latent vectors is an important problem to find the correspondences when the two or more graphs are from different domains. The optimal transport matrix and the embeddings are alternatively helped to update each other values. After getting the optimal transport matrix based on previous embedding values, we update the embedding values as in the minimization problem in (9), where  $\alpha_m$  is a parameter determines the contributions of the latent vectors to the proposed method. The value of  $\alpha_m$  starts from very small value, as the initial vectors are random. It increases in each iteration, as the latent vectors become more accurate to contribute in improving the matching.  $D(E_s, E_t)$  is a distance matrix between the embeddings and  $\hat{T}^{(m)}$  is an optimal transport matrix obtained based on the previous embeddings.

$$\min_{E_s, E_t} \alpha_m \langle D(E_s, E_t), \hat{T}^{(m)} \rangle \quad (9)$$

### 3.6. Graph matching

Graph matching (GM) refers to finding the correspondences between graphs. It plays an important role in many applications in different domains. In this work, we are interested in matching discrete descriptors of multi-objective dynamics such as matching descriptors of perimeter monitoring with and without the presence of an obstacle. The pseudo-code in (1) illustrates the main steps for obtaining optimal transport matrix, latent vectors and graph matching.

### 3.7. Sub-graph extraction

A threshold is applied to the optimal transport matrices to obtain the sub-graphs that are correspondence to the ab-

---

**Algorithm 1** Graph Matching based-on Gromov-Wasserstein
 

---

```

1: Input:  $\{C_s, C_t\}, \{\mu_s, \mu_t\}, \gamma, \{M, N, J\}$ 
2:  $E_s^{(0)}, E_t^{(0)} \leftarrow$  random values
3:  $\hat{T}^{(0)} \leftarrow \mu_s \mu_t^\top$ 
4:  $a \leftarrow \mu_s$ 
5: for  $m = 0$  to  $M - 1$  do
6:    $\alpha_m \leftarrow \frac{m}{M}$ 
7:   for  $n = 0$  to  $N - 1$  do
8:      $C^{(m,n)} = L(C_s, C_t, T^{(n)})$ 
9:      $\alpha D(E_s^{(m)}, E_t^{(m)}) + \gamma$ 
10:     $G \leftarrow \exp(-\frac{C^{(m,n)}}{\gamma}) \otimes T^{(n)}$ 
11:    for  $j = 1$  to  $J$  do
12:       $b \leftarrow \frac{\mu_t}{G^\top a}$ 
13:       $a \leftarrow \frac{\mu_s}{G b}$ 
14:     $T^{(n+1)} \leftarrow \text{diag}(a) G \text{diag}(b)$ 
15:   $\hat{T}^{(m+1)} \leftarrow T^{(N)}$ 
16:  Obtain  $E_s^{(m+1)}, E_t^{(m+1)}$  by solving (9)
17:  $E_s \leftarrow E_s^{(M)}, E_t \leftarrow E_t^{(M)}, \hat{T} \leftarrow \hat{T}^{(M)}$ 
18:  $GM \leftarrow \emptyset$ 
19: for  $v_i \in V_s$  do
20:    $j \leftarrow \text{argmax}_j \hat{T}_{ij}$ 
21:    $GM \leftarrow GM \cup \{(v_i \in V_s, v_j \in V_t)\}$ 
22: for  $v_j \in V_t$  do
23:    $i \leftarrow \text{argmax}_i \hat{T}_{ij}$ 
24:   if  $(v_i, v_j) \notin GM$  then
25:      $GM \leftarrow GM \cup \{(v_i \in V_s, v_j \in V_t)\}$ 
26: return  $E_s, E_t, \hat{T}$  and  $GM$ 

```

---

normalities.

The following subsections explain the exploited datasets, how to obtain the discrete descriptors and the transition matrices between these descriptors.

### 3.8. Datasets

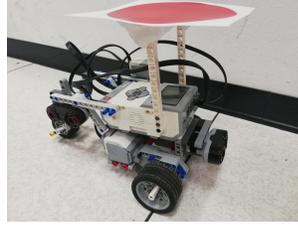
#### 3.8.1 Autonomous Systems

Two autonomous systems are employed to collect the dataset. The employed system architectures are shown in Fig. 2 where:

A small robot (Lego Mindstorms EV3) inside a simulated environment performed two scenarios. For monitoring the robot's state, three types of sensors are used:

- Odometry, which consists of the two motor encoders.
- Sonars, which collect the distance measurements.
- External camera placed on top of the scene, to extract the position of the robot in each frame.

A real vehicle called iCab [22] are used to collect a multi-sensory data.



(a) Lego Robot



(b) iCab Vehicle

Figure 2. Two different architectures are employed

#### 3.8.2 Scenarios

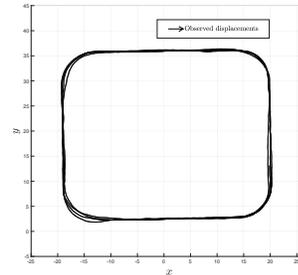
Different tasks are performed by the robot and the vehicle. Fig. 3 shows the two tasks that used in this paper:



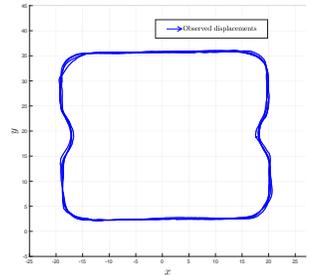
(a)



(b)



(c)



(d)

Figure 3. Two different scenarios for both datasets: (a,c) perimeter monitoring, (b,d) perimeter monitoring in the presence of an obstacle.

- **Perimeter monitoring (PM)** where the autonomous system follows a squared path inside the proposed environment.
- **Obstacle avoidance (OA)** where the system conducts perimeter monitoring until facing an obstacle. In this case, the system performs avoidance maneuver to avoid the obstacle, then continue performing the perimeter monitoring task.

### 3.9. Descriptors of data from same sensor with different tasks

Fig.4 shows the robot's positions (in blue) during performing two different tasks (perimeter monitoring with and without the presence of an obstacle). The yellow circles represent the descriptors and the black edges represent the connections between them. These circles and edges are generated by the clustering algorithm.

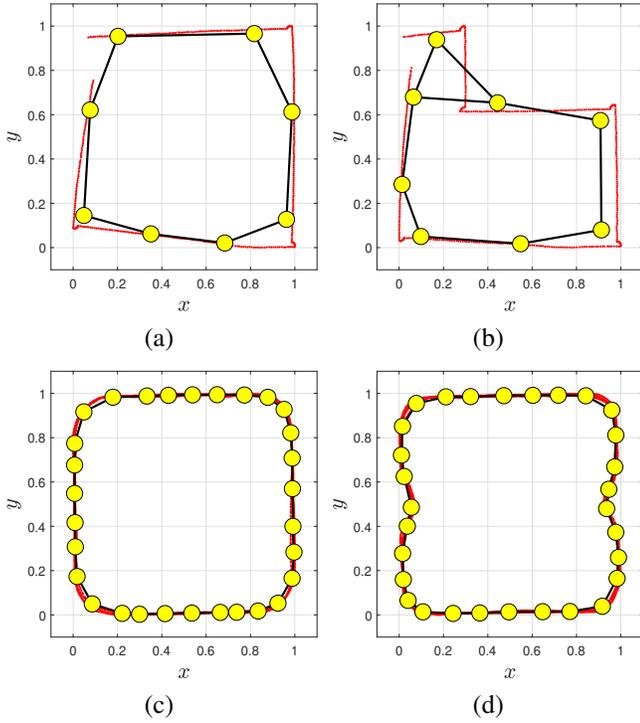


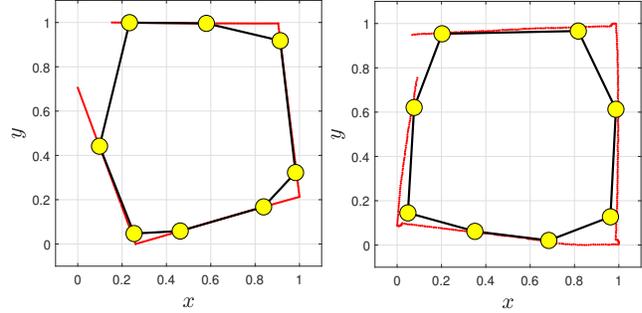
Figure 4. Two different scenarios for robot on top and iCab in bottom: (a,c) perimeter monitoring, (b,d) perimeter monitoring in the presence of an obstacle.

### 3.10. Descriptors of data from different sensors

Fig. 5 shows the robot's positions (in blue) from two different sensors (Odometry and external camera) during performing perimeter monitoring task. As mentioned before, the yellow circles represent the descriptors, which are generated by the GNG.

## 4. EXPERIMENTAL RESULTS

The following subsections present our results, which demonstrate the principle idea with results on synthetic data and we show the results on real data. For illustration purpose, we plot the graphs with different scales.



(a) via Odometry (b) via External Camera

Figure 5. Robot's positions from two different sensors

### 4.1. Synthetic data

To demonstrate proof of concept of matching graph based on transition probabilities and interactions, two synthetic examples are implemented. These examples illustrate that the performed method matches directed graphs correctly even when they have the same values of transition probabilities, but the directions are different. In the first example, we consider the transition matrices for the source and target graphs, which we want to match are the same as  $TransMat1$ . While in the second example, we consider the transition matrix for the source graph is similar to  $TransMat1$  and for the target graph is similar to  $TransMat2$ . Fig.6 illustrates the graphs matching with their optimal transport matrices, where the black lines represent the edges within the same graph, while the colored lines represent the correspondences from different graphs.

$$TransMat1 = \begin{pmatrix} a1 & b1 & c1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{matrix} a1 \\ b1 \\ c1 \end{matrix}$$

$$TransMat2 = \begin{pmatrix} a2 & b2 & c2 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \begin{matrix} a2 \\ b2 \\ c2 \end{matrix}$$

To validate the proposed method, we provided the same graph as source and target and we check the matching result. Fig.7 illustrates that.

### 4.2. Real data

Fig.8 shows the graph matching of obstacles avoidance scenario (inner graph) with perimeter monitoring one (outer graph). Also, it presents the optimal transport matrix between the correspondences from both graphs. Here, the order of the nodes in rows and columns of transport and transition matrices are based on the clustering algorithm. While

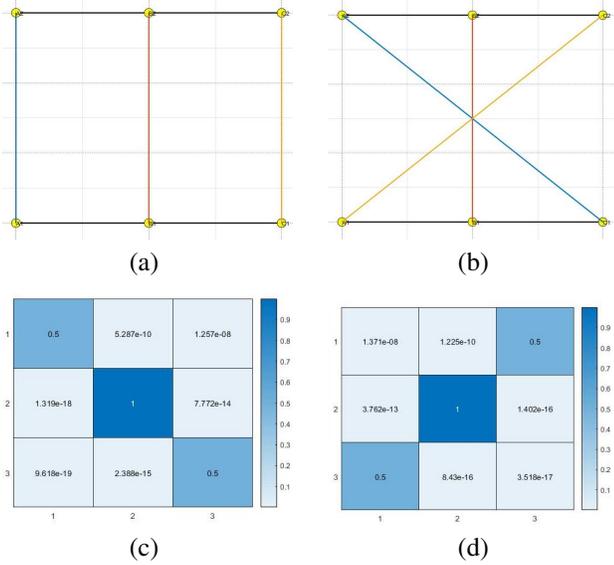


Figure 6. Matching of directed graphs with their optimal transport matrices.

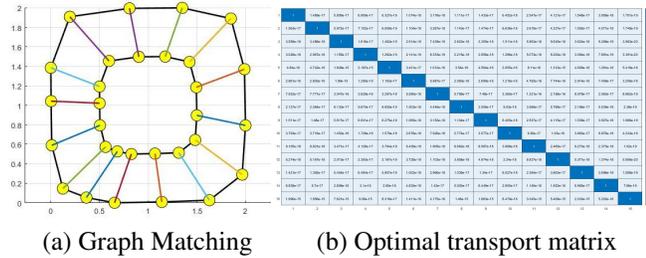


Figure 7. Synthetic validation

Fig.9 presents matching graphs that are collected by different sensors for perimeter monitoring with and without the presences of an obstacle. In the figure, we can see some matching could not be desirable. This could be explained due to depending only on statistical information, where it matched nodes that have similar information in different places within the same graph.

### 4.3. Extracting Sub-graph of different Experiences

Fig.10 presents the results of extracting sub-graphs that are related to the differences between different experiences are collected from Lego robot. It also shows sub-graph based-on graph matching of descriptors which are obtained from different sensors. Also, Fig.11 presents the results of extracting sub-graphs that are related to the differences between perimeter monitoring with and without obstacles avoidance from iCAB car. A threshold is applied to the optimal transport matrices to obtain these sub-graphs. This threshold is selected empirically, where in this work it is selected based-on the mean and variance of the source graph's node. To have a general formula the for the required thresh-

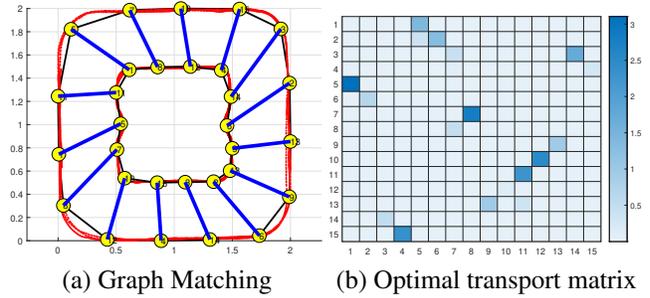


Figure 8. Matching graphs of perimeter monitoring (outer graph) with obstacles avoidance (inner graph)

old, future work will be conducted with dataset contains enough scenarios.

### 4.4. Limitations

In this paper, we have used some statistical information about the graphs. We used in each graph, the transition matrix between the nodes and the mean of each node. In addition, in the current scenarios, almost every node is just connected with two neighbors. These lead to problems in matching graphs with a high number of nodes. It also has a

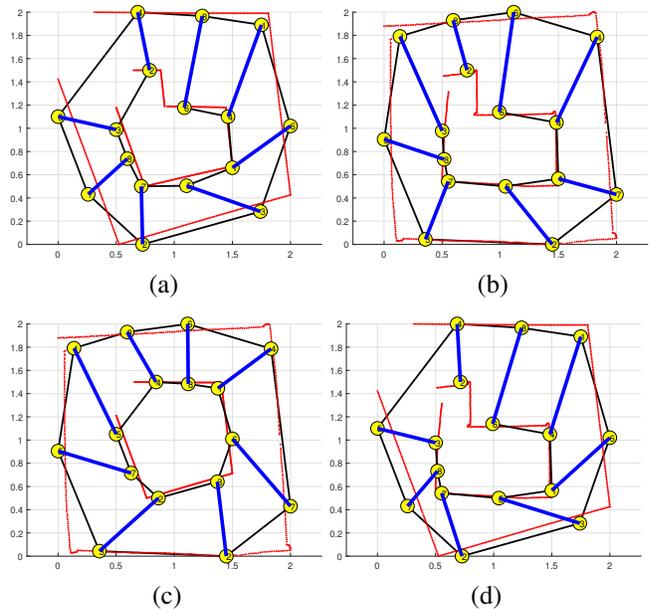


Figure 9. Matching graphs as following: (a) is perimeter monitoring (outer graph) with obstacles avoidance (inner graph) where both are based-on Odometry, (b) is perimeter monitoring (outer graph) with obstacles avoidance (inner graph) where both are based-on external camera, (c) is perimeter monitoring (inner graph) from Odometry with perimeter monitoring (outer graph) from the external camera, (d) is perimeter monitoring (outer graph) from Odometry with obstacles avoidance (inner graph) from external camera.

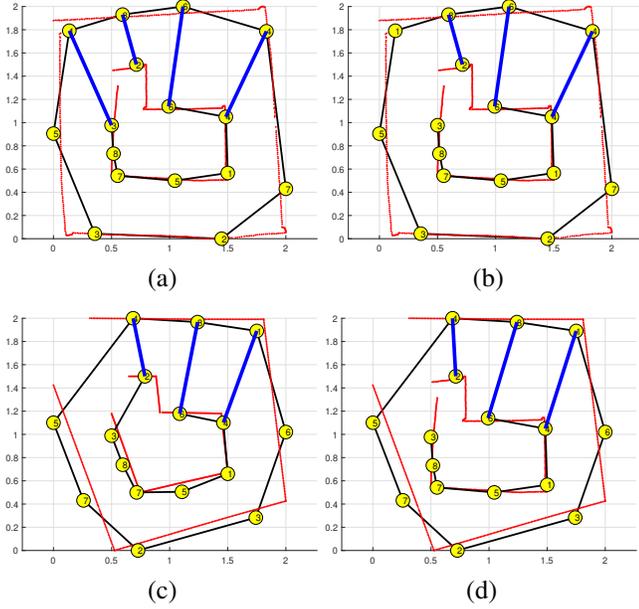


Figure 10. Matching sub-graphs associated with area of different experiences. (a,b) are sub-graph matching in for perimeter monitoring (outer graph) with obstacles avoidance (inner graph) where both based-on external camera, with different threshold values, (c) is sub-graph matching for perimeter monitoring (outer graph) with obstacles avoidance (inner graph) where both are based-on Odometry, (d) is sub-graph matching for perimeter monitoring (outer graph) from Odometry with obstacles avoidance (inner graph) from external camera.

problem when the number of nodes is big and the dissimilarity between graphs is small, which makes it challenging to extract abnormality. These issues are required an investigation of some parameters to to improve the matching and extracting the sub-graphs. These parameters are the optimal number of nodes are generated from clustering stage for better transition matrices and the threshold value is required for extracting the sub-graphs.

## 5. CONCLUSION AND FUTURE WORK

This paper has investigated graph matching of the abstract states of multi-tasks dynamics. We have managed to match graphs which are generated from different sensors by different maneuvering tasks such as perimeter monitoring with and without the presence of an obstacle. Gromov-Wasserstein framework is used for this purpose, which measures the distance between samples in each domain, then it compares these distances. This has an advantage when the two graphs from different domains with different dimensions. In addition, latent vectors have jointly learned to improve finding the best correspondence. After obtaining the optimal transport matrix, a threshold is applied to extract sub-graph that is associated with the abnormality. Such sub-

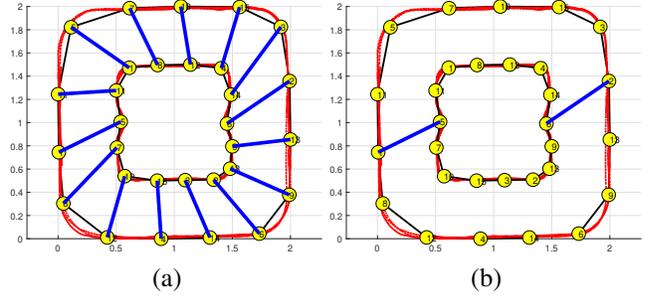


Figure 11. (a) shows graph matching of perimeter monitoring with and without obstacles avoidance and (b) shows matching sub-graphs associated with area of different experiences.

graph could be used as a micro-skill for incremental learning and transfer knowledge.

Future work will investigate including more information about the dynamics of the descriptors to enhance graph matching of big graphs. Also to increase the number of scenarios and use full images instead of extracting the position from a third person camera. To further our research, we are planning to exploit the idea of using the extracted sub-graphs to build a dictionary of micro-skills to transfer them between several agents.

## References

- [1] Johannes Schlatow, Mischa Möstl, Rolf Ernst, Marcus Nolte, Inga Jatzkowski, Markus Maurer, Christian Herber, and Andreas Herkersdorf. Self-awareness in autonomous automotive systems. In *Proceedings of the Conference on Design, Automation & Test in Europe*, pages 1050–1055. European Design and Automation Association, 2017.
- [2] Ryan Elwell and Robi Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, 2011.
- [3] Zhi Wang, Chunlin Chen, Hanxiong Li, Daoyi Dong, and Tzyh Jong Tarn. Incremental reinforcement learning with prioritized sweeping for dynamic environments. *IEEE/ASME Transactions on Mechatronics*, 2019.
- [4] Mahdyar Ravanbakhsh, Mohamad Baydoun, Damian Campo, Pablo Marin, David Martin, Lucio Marcenaro, and Carlo S Regazzoni. Learning multi-modal self-awareness models for autonomous vehicles from human driving. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 1866–1873. IEEE, 2018.
- [5] Matheus RF Mendonça, Artur Ziviani, and André Barreto. Graph-based skill acquisition for reinforcement learning. *ACM Computing Surveys (CSUR)*, 52(1):6, 2019.
- [6] Matheus Ribeiro Furtado de Mendonça, Artur Ziviani, and André da Motta Salles Barreto. Abstract state transition graphs for model-based reinforcement learning. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 115–120. IEEE, 2018.
- [7] Jaekoo Lee, Hyunjae Kim, Jongsun Lee, and Sungroh Yoon. Transfer learning for deep learning on graph-structured data.

- In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [8] Yifeng Cai and Kosuke Sekiyama. Subgraph matching route navigation by uav and ground robot cooperation. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 4881–4886. IEEE, 2016.
- [9] Daisuke Kakuma, Satoki Tsuchihara, Gustavo Alfonso Garcia Ricardez, Jun Takamatsu, and Tsukasa Ogasawara. Alignment of occupancy grid and floor maps using graph matching. In *2017 IEEE 11th international conference on semantic computing (ICSC)*, pages 57–60. IEEE, 2017.
- [10] Andrei Zanfir and Cristian Sminchisescu. Deep learning of graph matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2684–2693, 2018.
- [11] Elif Vural. Domain adaptation on graphs by learning graph topologies: Theoretical analysis and an algorithm. *arXiv preprint arXiv:1812.06944*, 2018.
- [12] Facundo Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.
- [13] Junchi Yan, Xu-Cheng Yin, Weiyao Lin, Cheng Deng, Hongyuan Zha, and Xiaokang Yang. A short survey of recent advances in graph matching. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, pages 167–174. ACM, 2016.
- [14] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 25–32, 2013.
- [15] Mohamad Baydoun, Damian Campo, Valentina Sanguineti, Lucio Marcenaro, Andrea Cavallaro, and C Regazzoni. Learning switching models for abnormality detection for autonomous driving. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 2606–2613. IEEE, 2018.
- [16] Mahdyar Ravanbakhsh, Enver Sangineto, Moin Nabi, and Nicu Sebe. Training adversarial discriminators for cross-channel abnormal event detection in crowds. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1896–1904. IEEE, 2019.
- [17] Mohamad Baydoun, Mahdyar Ravanbakhsh, Damian Campo, Pablo Marin, David Martin, Lucio Marcenaro, Andrea Cavallaro, and Carlo S Regazzoni. A multi-perspective approach to anomaly detection for self-aware embodied agents. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6598–6602. IEEE, 2018.
- [18] Bernd Fritzke. A growing neural gas network learns topologies. In *Advances in neural information processing systems*, pages 625–632, 1995.
- [19] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin. Gromov–wasserstein learning for graph matching and node embedding. *arXiv preprint arXiv:1901.06003*, 2019.
- [20] Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. A fast proximal point method for computing wasserstein distance. *arXiv preprint arXiv:1802.04307*, 2018.
- [21] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013.
- [22] Pablo Marin-Plaza, Jorge Beltrán, Ahmed Hussein, Basam Musleh, David Martin, Arturo de la Escalera, and José María Armingol. Stereo vision-based local occupancy grid map for autonomous navigation in ros. In *11th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications VISIGRAPP*, volume 2016, 2016.