2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Graduated Assignment Graph Matching for Realtime Matching of Image Wireframes

Joseph Menke[1] and Allen Y. Yang[2]

***Abstract*— We present an algorithm for the realtime matching of wireframe extractions in pairs of images. Here we treat extracted wireframes as graphs and propose a simplified Graduated Assignment algorithm to use with this problem. Using this algorithm we achieve a 30% accuracy improvement over the baseline method. We show that, for this problem, the simplified Graduated Assignment algorithm can achieve realtime performance without a significant drop in accuracy as compared to the standard Graduated Assignment algorithm. We further demonstrate a method of utilizing this simplified Graduated Assignment algorithm for achieving a similar realtime improvement in the matching quality of standard features without wireframe detection.**

I. INTRODUCTION

3D reconstruction is an important task for many computer vision applications such as robotics and augmented reality. While computing power continues to increase, the burden of collecting data remains a fixed cost for the user. As such, many are looking towards reducing the data required at runtime to generate accurate reconstructions. One method to achieve this is to use large amounts of data, collected ahead of time, to provide a strong prior on how data collected at runtime should be interpreted [1], [2], [3], [4], [5]. In the extreme case this allows depth reconstruction from a single monocular image [6], [7], [8]. However these approaches can be fragile to minor changes in camera parameters such as angle or focal length [9], [10] and overcoming these issues may require an intractable amount of data. Instead we may look to spend additional computation to ensure we make the most out of the data we collect at runtime, thereby reducing the need to collect redundant data.

Due to the inherent ambiguity in visual features, a significant amount of information is discarded due to an inability to accurately associate these features across images. Graph matching has been shown to be successful as a method for comparing images [11], [12], [13] or matching features in images [14], however, due to the high computation cost, it has only been applied in cases with relatively few (≈ 40) nodes. If realtime performance can be achieved, this potentially provides us with a better way to associate structures in an image, and reduce the amount of information lost.

In this paper we combine the above approaches, utilizing a neural network trained to identify important structures (ie. wireframes) in the scene [15], and then apply Graduated Assignment graph matching [16] to match these structures between images, achieving improved performance over baseline feature matching. We further show that our simplification of Graduated Assignment is able to achieve realtime performance (30 FPS) on graphs with up to 300 nodes without a loss in accuracy compared to Graduated Assignment. We show that this approach can also be applied to achieve improved performance in standard feature matching without the use of a wireframe.

*Research supported in part by ONR under grant N00014-19-1-2055

[1]Joseph Menke is a Graduate Student of Electrical Engineering and Computer Science University of California at Berkeley, Berkeley, CA 94720 joemenke@eecs.berkeley.edu

[2]Allen Y. Yang is an Associate Professor of Electrical Engineering and Computer Science University of California at Berkeley, Berkeley, CA 94720 yang@eecs.berkeley.edu

II. RELATED WORKS

A. Wireframe Detection

A wireframe consists of a set of straight lines and their intersected junctions and is analogous to line drawings commonly used in architecture design [17]. These lines and junctions represent some of the most fundamental elements that can be used to infer the geometric structure of the scene. They are also a compact representation, using far less information to describe the scene than point clouds or occupancy grids.

Wireframes can be detected using a simple line segment detector [18] and the junctions inferred through some simple heuristics [19]. Huang et al. [17] showed that these wireframes can be more accurately detected using a neural network trained on a well labeled dataset. Zhou et al. [15] further improved these results by switching to an end to end parsing scheme. We utilize the publicly available network from Zhou et al. [15] in our results going forward, however our method for wireframe matching is not dependent on the particular method of wireframe extraction used. An example wireframe detection is shown in Figure 1. Notice that the wireframe typically involves a sparse set of connections between junctions. That is, a given junction only has a few lines associated with it and is only directly connected with a small number of other junctions.

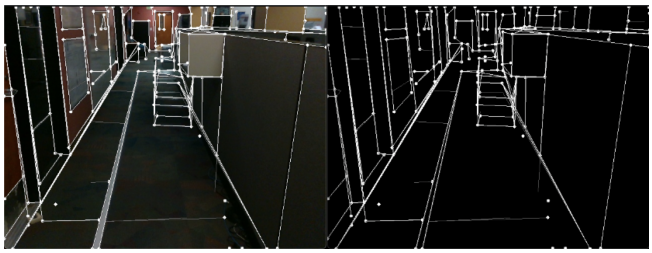

Fig. 1. Example wireframe detection result overlayed on the RGB image (left) and overlayed on a black image (right).

B. Graph Matching

As we discuss later, a wireframe can be considered a graph linking visual features in an image. Therefore, it is reasonable to consider wireframe matching in images as a special case of graph matching. Graph matching is a very difficult problem with many applications. Several different methods are employed that can achieve approximate solutions to the general problem. The most basic of these is the Hungarian algorithm [20] which solves the assignment problem given a measure of similarity of a variable to each assignment. This has been applied by several groups to give an approximate solution to the graph matching problem under various measures of similarity between nodes [21], [22], [23], [24].

While these algorithms can work well given a good similarity metric, the majority of these algorithms are "local" algorithms meaning they can only compare the similarity of a region around a given node. This has issues if there are repeated structures in the graph. More advanced approaches look to solve an optimization problem (shown in Section IV) that maximizes the overall coherence of the graphs [16], [25], [26], [27], [28], [29], [14], [30]. Of these, the Graduated Assignment algorithm [16] has been used extensively in computer vision especially in matching image skeletons [11], [13], [12].

More recently, several algorithms have been shown to outperform Graduated Assignment [25], [26], [27], [29], [14]. Unfortunately these papers utilize an open source implementation [25] of Graduated Assignment that does not have good performance under the parameters defined in the original paper. Specifically, the open-source implementation performs a scaling on the similarity matrix that causes the algorithm to take very small steps at each iteration. While we cannot re-implement all the above algorithms, removing this scaling causes significant improvement to the Graduated Assignment algorithm for tests made by Zhou and De la Torre [27], leaving it on par with the best algorithms evaluated in the paper. See Appendix for results.

Neural Networks have also been applied to the task of Graph Matching [31], [32]. The most relevant result here is the work of Aboudib et al. [31] that extends an iterative optimization approach via the incorporation of neural networks. It is likely that these methods will outperform the results in this paper in terms of accuracy under the condition that the test environment is well represented in the training set. This will be at the cost of increased computation and additional prior data collection.

C. Feature Matching

The matching of wireframes can also be considered a special case of standard feature matching where the visual similarity of either the lines or junctions is used to match these features across images. Many methods exists that look at robustly describing the image patch around the feature in a way that can be easily compared across images [33], [34], [35], [36], [18], [37]. These approaches establish a distance metric between features, such as Hamming distance for binary descriptors [33]. A simple nearest neighbor matching approach is to match each feature with its closest feature under this metric. This is what we term "standard feature matching" in this paper. While these methods work very well, these features suffer greatly in the presence of repeated textures; a common occurrence in man-made environments.

Similar in spirit to our approach is that of graph cuts [38] which can be used to efficiently find a labeling of pixels that minimizes an energy function. This has a wide variety of applications in the field of computer vision, but most relevant here is the application to stereo feature matching [39] and multi-view feature matching [40]. Rather than matching sparse features directly, these methods attempt to assign a depth or disparity label to each pixel. This requires a known transformation between images, which we may not have at the time of matching. It also requires a fixed number of disparity labels limiting the resolution of the reconstruction. Note that, while graph cuts can solve a wide variety of energy minimization problems, known as submodular functions [41], graph matching does not fall into this category [14].

III. Contribution

The contributions of this paper are as follows:

1) We propose a simplification of the Graduated Assignment algorithm that, for certain tasks, can achieve realtime performance without a significant reduction in accuracy. Tasks involving 3D maps, such as path planning and obstacle avoidance, often require realtime performance but, until now, this type of matching algorithm has been outside realtime performance for many scenarios.
2) We apply this simplified Graduated Assignment algorithm to the task of wireframe matching by utilizing the visual similarity of junctions and lines. This represents an important step towards automatic 3D wireframe reconstruction, which will enable compact models that can be used in robotics and augmented reality. We further show how this method can be adapted for general realtime feature matching.

IV. Problem Statement

A wireframe can easily be represented as a graph, where each junction represents a node of the graph, and each line represents an edge. To optimize matching of wireframes in images, we need to define the metric to optimize over. The simplest metric is to maximize matching nodes whose connected nodes are also connected in the corresponding image. This is a standard graph matching problem:

$$\max_{\mathbf{M}} S^{\text{gm}}(\mathbf{M}) = \sum_{a=1}^{n_1}\sum_{i=1}^{n_2}\sum_{b=a+1}^{n_1}\sum_{j=i+1}^{n_2} \mathbf{C}_{abij}\mathbf{M}_{ai}\mathbf{M}_{bj}, \quad (1)$$

$$\text{s.t. } \forall a \sum_{i=1}^{n_2} \mathbf{M}_{ai} \leq 1,\ \forall i \sum_{a=1}^{n_1} \mathbf{M}_{ai} \leq 1,\ \forall ai\, \mathbf{M}_{ai} \in \{0,1\},$$

where S^{gm} is the similarity function we would like to maximize. If we denote the two graphs $\mathbf{G}_1$ and $\mathbf{G}_2$ then n_1

and n are the number of nodes in $\mathbf{G}_1$ and $\mathbf{G}_2$ respectively. We can represent $\mathbf{G}_1$ and $\mathbf{G}_2$ as sparse matrices where $\mathbf{G}_{1ab} = 1$ if node a is connected to node b and 0 otherwise. $\mathbf{G}_{2ij}$ follows similarly. $\mathbf{C}_{abij}$ represents the compatibility of these edges. That is:

$$\mathbf{C}_{abij} = \begin{cases} 1 & \mathbf{G}_{1ab}\mathbf{G}_{2ij} = 1 \\ 0 & otherwise \end{cases}$$

The Matrix $\mathbf{M}$ represents the correspondence of nodes, with $\mathbf{M}_{ai} = 1$ indicating the matching of node a in $\mathbf{G}_1$ to node i in $\mathbf{G}_2$. The inequality constraints in this optimization enforce that a node in $\mathbf{G}_1$ can only be matched to at most one node in $\mathbf{G}_2$. It is important to note here that the matrix C need not be computed directly. If $\mathbf{G}_1$ and $\mathbf{G}_2$ are represented as sparse matrixes, the similarity function S^{gm} can be computed very efficiently in $O(l_1 l_2)$ time (where l_1 and l_2 are the number of edges in $\mathbf{G}_1$ and $\mathbf{G}_2$ respectively) by iterating through the non-zero elements of $\mathbf{G}_1$ and $\mathbf{G}_2$.

An alternative method to the above would be to additionally maximize the visual similarity of the nodes:

$$\max_{\mathbf{M}} S^{\text{agm}}(\mathbf{M}) = \sum_{a=1}^{n_1}\sum_{i=1}^{n_2}\sum_{b=a+1}^{n_1}\sum_{j=i+1}^{n_2} \mathbf{C}_{ab,ij}\mathbf{M}_{ai}\mathbf{M}_{bj} + \alpha \sum_{a=1}^{n_1}\sum_{i=1}^{n_2} \mathbf{\Theta}^{\text{node}}_{ai}\mathbf{M}_{ai}, \quad (2)$$

$$\text{s.t. } \forall a \textstyle\sum_{i=1}^{n_2} \mathbf{M}_{ai} \leq 1,\ \forall i \sum_{a=1}^{n_1} \mathbf{M}_{ai} \leq 1,\ \forall ai\, \mathbf{M}_{ai} \in \{0,1\},$$

where $\mathbf{\Theta}^{\text{node}}_{ai}$ represents the visual similarity of the node a in $\mathbf{G}_1$ to node i in $\mathbf{G}_2$. α is a weighting parameter to trade off between the importance of the two tasks. This becomes an attributed graph matching problem.

Lastly we could go further and maximize the visual similarity of the edges connecting the nodes:

$$\max_{\mathbf{M}} S^{\text{awgm}}(\mathbf{M}) = \sum_{a=1}^{n_1}\sum_{i=1}^{n_2}\sum_{b=a+1}^{n_1}\sum_{j=i+1}^{n_2} \mathbf{\Theta}^{\text{edge}}_{abij}\mathbf{M}_{ai}\mathbf{M}_{bj} + \alpha \sum_{a=1}^{n_1}\sum_{i=1}^{n_2} \mathbf{\Theta}^{\text{node}}_{ai}\mathbf{M}_{ai}, \quad (3)$$

$$\text{s.t. } \forall a \textstyle\sum_{i=1}^{n_2} \mathbf{M}_{ai} \leq 1,\ \forall i \sum_{a=1}^{n_1} \mathbf{M}_{ai} \leq 1,\ \forall ai\, \mathbf{M}_{ai} \in \{0,1\},$$

where $\mathbf{\Theta}^{\text{edge}}_{abij}$ represents the visual similarity of the edge between nodes a and b in $\mathbf{G}_1$ to the edge between nodes i and j in $\mathbf{G}_2$. $\mathbf{\Theta}^{\text{edge}}_{abij}$ will only be non-zero when $\mathbf{C}_{abij}$ is non-zero. Similar to Equation 1, we do not need to compute $\mathbf{\Theta}^{\text{edge}}$ directly. Instead, we make the sparse matrices $\mathbf{G}_1$ and $\mathbf{G}_2$ contain the index of an edge in a edge similarity matrix of size l_1 x l_2 and once again iterate through the non-zero elements of $\mathbf{G}_1$ and $\mathbf{G}_2$. Equation 3 is a form of attributed weighted graph matching.

As all of the above are generalizations of the inexact graph matching problem, they, unfortunately, fall into the class of NP-Hard problems [14]. As such, we look to find an approximate solution to this problem.

V. Algorithm Overview

Graduated Assignment is well described by Gold and Rangarajan [16] and has been shown to be capable of providing an approximate solution to the problem above. While Graduated Assignment is very efficient with a complexity of $O(l_1 l_2)$, where l_1 and l_2 are the number of edges in $\mathbf{G}_1$ and $\mathbf{G}_2$ respectively, the number of iterations required can still induce a high computation cost. We motivate a faster algorithm by looking at what reductions can be made to Graduated Assignment for the wireframe matching task. In this section we describe our simplified solution as it compares to the original algorithm. While the simplifications have the potential to reduce the accuracy of the final matching, we attempt to motivate why this accuracy reduction should be minimal. The solution here is described specifically as applied to Equation 3, however the solutions for Equations 1 and 2 follow similarly.

First we define the energy function as the negative of the similarity function ($E^{\text{awgm}} = -S^{\text{awgm}}$) and maximize the similarity function by minimizing the energy function. We follow the method of the Graduated Assignment algorithm by defining an initial correspondence matrix $\mathbf{M}^0$ and taking the first order Taylor Series approximation of this energy function:

$$E^{\text{awgm}}(\mathbf{M}) \approx E^{\text{awgm}}(\mathbf{M}^0) - \sum_{a=1}^{n_1}\sum_{i=1}^{n_2} \mathbf{Q}_{ai}(\mathbf{M}^0_{ai} - \mathbf{M}_{ai}) \quad (4)$$

$$\mathbf{Q}_{ai} = \alpha\mathbf{\Theta}^{\text{node}}_{ai} + \sum_{b=a+1}^{n_1}\sum_{j=i+1}^{n_2} \mathbf{\Theta}^{\text{edge}}_{abij}\mathbf{M}^0_{bj} \quad (5)$$

From this we find that we can descend on our energy function by maximizing:

$$\sum_{a=1}^{n_1}\sum_{i=1}^{n_2} \mathbf{Q}_{ai}\mathbf{M}_{ai}, \quad (6)$$

By finding the $\mathbf{M}$ that maximizes (6) and repeatedly relinearizing about our new $\mathbf{M}$ we can repeatedly descend on this function until we achieve a local optimum. This, however, is still non-trivial due to the constraints we impose on $\mathbf{M}$ in Equation 3.

A. The Set Constraint

As with the original algorithm, to deal with the constraint $\mathbf{M}_{ai} \in \{0,1\}$, we employ a continuation method known as simulated annealing or graduated non-convexity [42]. This involves relaxing the constraint to allow the elements of $\mathbf{M}$ to lie in the continuous range $[0,1]$. A control variable β is used to slowly push the values of $\mathbf{M}$ to either 0 or 1. This is done by utilizing the softmax function:

$$x'_i = \frac{e^{\beta x_i}}{\sum_{j=1}^{n} e^{\beta x_j}} \tag{7}$$

By increasing β over time we approximate our original constraint. In Graduated Assignment, multiple iterations are used to descend on $E^{\text{awgm}}(\mathbf{M})$ for each value of β. For several reasons, this may not be necessary. First, we don't actually need to converge for a given value of β, we just need to get to a value of $\mathbf{M}$ where the next β still leaves us in a good local concavity. Second, we don't necessarily need to converge to the minima at the last step either, as we can instead perform a greedy assignment step to fully enforce the constraint $\mathbf{M}_{ai} \in \{0, 1\}$. This involves taking the max of each row in $\mathbf{M}$, setting it to 1, and setting all other elements of $\mathbf{M}$ to 0. Lastly, we note that as the graph becomes more sparse, the number of nonlinear terms decreases, thus reducing the need for multiple iterations. As wireframes typically form very sparse graphs, we propose that in performing a single descent step for each value of β we do not sacrifice significant accuracy as compared to the iterative approach.

B. The Uniqueness Constraint

The inequality constraints on $\mathbf{M}$ ($\forall a \sum_{i=1}^{n_2} \mathbf{M}_{ai} \leq 1$, $\forall i \sum_{a=1}^{n_1} \mathbf{M}_{ai} \leq 1$) enforce that at each node in $\mathbf{G}_1$ is uniquely matched to, at most, a single node in $\mathbf{G}_2$. One way to enforce these constraints is to first note that if these were equality constraints ($\forall a \sum_{i=1}^{n_2} \mathbf{M}_{ai} = 1$, $\forall i \sum_{a=1}^{n_1} \mathbf{M}_{ai} = 1$) and $\mathbf{M}$ were a square matrix then $\mathbf{M}$ would be what is known as a doubly stochastic matrix. It is well known that any square matrix with all positive elements can be converted into a doubly stochastic matrix via the Sinkhorn-Knopp algorithm [43]. This involves alternating between normalizing all rows and normalizing all columns until the algorithm converges. Thus, if our matrix is square, we can simply add an additional row and column of "slack variables", and by applying the Sinkhorn-Knopp algorithm, we enforce our inequality constraint on our original matrix.

Graduated Assignment uses the Sinkhorn-Knopp algorithm with slack variables to both enforce the uniqueness constraints and determine outliers (the issue of non-square matrices is not addressed). For image feature matching there exist many good methods for determining outliers. As such, we propose only enforcing the constraints unilaterally and apply external outlier rejection to remove non-unique feature matches. It is noted by Gold and Rangarajan [16] that performing only one iteration of the Sinkhorn-Knopp algorithm is identical to only enforcing the constraint in one direction. This also removes the need for slack variables. It is worth noting that each Sinkhorn-Knopp iteration is only $O(n_1 n_2)$ rather than $O(l_1 l_2)$. That is, it's complexity grows with the number of nodes in the graphs rather than the number of edges. In a densely connected graph this change does not reduce the computational cost significantly as the number of edges would far exceed the number of nodes. As mentioned above, however, wireframes in a man-made environment are likely to be very sparse meaning that this step makes up a significant part of the algorithm's computation cost.

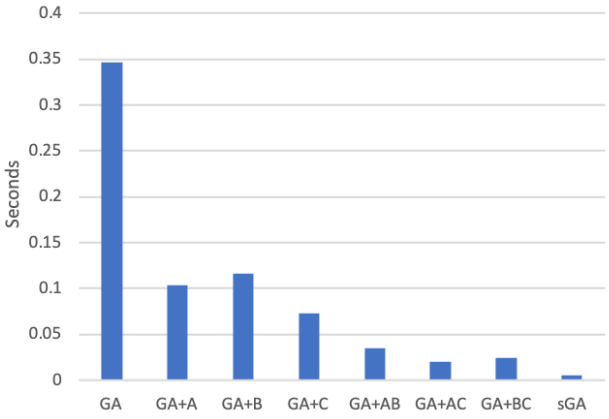

Fig. 2. Effect of each of the simplifications from section V on the average runtime of wireframe matching on fr3. sGA is the full simplified algorithm with all the changes applied. "A" are the reductions proposed in the "Set Constraint" subsection. "B" are the reductions proposed in the "Uniqueness Constraint" subsection. "C" are the reductions proposed in the "Additional Parameters" subsection.

C. Additional Parameters

The recommended parameters for the original Graduated Assignment, given by Gold and Rangarajan [16], are: the initial annealing parameter $\beta_0 = 0.5$, the final annealing parameter $\beta_f = 10$, the rate of increase $\beta_r = 1.075$, the maximum number of descent iterations for a given β $t_2^{\max} = 4$, and the maximum number of Sinkhorn-Knopp iterations for each descent $t_3^{\max} = 30$. There are several changes to these parameters for our algorithm.

First, both t_2 and t_3 are effectively set to 1 due to the reductions proposed above. Second, we note that the rate of our simulated annealing algorithm is typically intended to be small to attempt to push us slowly towards a global optimum. If we accept that we are only looking to improve on our energy function and not necessarily achieve a global optimum, then we can increase this parameter significantly as compared to the original algorithm. Thus we increase the parameter β_r to 1.5. Lastly for our work, we initialize $\mathbf{M}$ to the node similarity matrix $\mathbf{\Theta}^{\text{node}}$. As we are not starting from an entirely random initialization of $\mathbf{M}$, we increase β_0 slightly to 1.0.

The full simplified Graduated Assignment is shown in Algorithm 1. The recommended parameters are the following: $\beta_0 = 1$, $\beta_f = 10$, $\beta_r = 1.5$.

These changes together have the effect of reducing the total worst case number of Sinkhorn-Knopp iterations (which are $O(n_1 n_2)$) by a factor of 1000, and the worst case number descent steps (which are $O(l_1 l_2)$) by a factor of 30 as compared to the original Graduated Assignment with recommended parameters. The effect of each of these changes on runtime is shown in Figure 2.

Algorithm 1: Simplified Graduated Assignment For AWGM

$\beta \leftarrow \beta_0$
$\mathbf{M} \leftarrow \mathbf{\Theta}^{\text{node}}$
while $\beta < \beta_f$ **do**
  $\forall a \in n_1 \, \forall i \in n_2$
  $\mathbf{Q}_{ai} \leftarrow \alpha \mathbf{\Theta}^{\text{node}}_{ai} + \sum_{b=a+1}^{n_1} \sum_{j=i+1}^{n_2} \mathbf{\Theta}^{\text{edge}}_{abij} \mathbf{M}^0_{bj}$
  -
  $\forall a \in n_1 \, \forall i \in n_2$
  $\mathbf{M}_{ai} \leftarrow e^{\beta \mathbf{Q}_{ai}}$
  -
  $\forall a \in n_1 \, \forall i \in n_2$
  $\mathbf{M}'_{ai} \leftarrow \frac{\mathbf{M}_{ai}}{\sum_{j=0}^{n_2+1} \mathbf{M}_{aj}}$
  -
  $\forall a \in n_1 \, \forall i \in n_2$
  $\mathbf{M}_{ai} \leftarrow \frac{\mathbf{M}'_{ai}}{\sum_{b=0}^{n_1+1} \mathbf{M}'_{bi}}$
  -
  $\beta \leftarrow \beta * \beta_r$
end
$\mathbf{M} \leftarrow$greedy_assignment$(\mathbf{M})$

VI. Applications

In this paper we look at two applications of our simplified Graduated Assignment. While our primary interest is in the matching of image wireframes, we understand that the cost of wireframe extraction (when specialized hardware is unavailable) would likely outweigh the benefits of having a realtime matching algorithm. As such, we are also interested in the applicability of the reduced Graduated Assignment to standard feature matching. The use in standard feature matching (for which we use ORB features), would allow this algorithm to potentially improve a wide variety of algorithms that use feature matching in their pipeline. These algorithms include Simultaneous Localization And Mapping and 3D object tracking.

A. Wireframe Matching

We utilize the publicly available, pretrained network from Zhou et al. [15] for wireframe extraction. In this network the number of detected junctions is set to 250 for each image, and we keep all lines with a score of 0.9 or greater up to a maximum of 2500 lines. While the network always produces 250 junctions, it is possible for these junctions to be in identical locations and so we remove duplicate junctions before proceeding. In order to support visual feature matching, we extract BREIF [33] descriptors around each junction, and binary line descriptors [18], [37] around each line. The similarity matrix $\mathbf{\Theta}^{\text{node}}$ is therefore computed by first finding the Hamming distance of the BRIEF descriptors from the first image to those from the second image. We truncate the distance to a maximum value $d^{\max} = 50$ for easier matrix scaling. This however gives us a measure of distance rather than a measure of similarity, and so we transform the matrix by: $\mathbf{\Theta}^{\text{node}} = (d^{\max} - \gamma^{\text{node}})/d^{\max}$ where γ^{node} is the matrix of descriptor distances. The reduced $(l_1 \, \text{x} \, l_2)$ line similarity matrix follows similarly. As both the line and feature matches are similarly scaled and measure similar information, we set the α parameter to 1.

B. ORB Feature Matching

For our standard feature matching tests, we first detect a maximum of 300 ORB [35] features in each image. To convert our set of features in each image into a graph, we add an edge between any two features whose distance in image space is less that τ pixels. We then optimize using the energy function defined in Equation 2. This has the effect of optimizing under the premise that features that are close in one image, are likely to be close in the second image. This is true whenever the scene is relatively static and the features are far from the image plane compared to the translation between the two images. For our experiments using ORB features we set τ to 10 pixels on 640x480 resolution images and only consider matches whose hamming distance is less than 50. We do not compute edge similarity for this experiment, however many others have proposed a wide variety of similarity measures [14], [26], [25], that could also be used in practice.

VII. Experiments

We compare the two versions of Graduated Assignment discussed in this paper. We denote the original method as "GA", and our simplified Graduated Assignment as "sGA". Our baseline comparison in both of the above cases is standard nearest neighbor feature matching (denoted "BF"). As this does not use any line information we compare against two versions of the above algorithms. The first which we denote with the suffix "_Node", only uses node similarity (solving Equation 2 rather than 3). The suffix "_Both" denotes the use of both node and edge similarity. As we do not compute edge similarity for ORB feature matching, all the algorithms used in that experiment use only node similarity.

For Graduated Assignment we employ a slight deviation from the algorithm described by Gold and Rangarajan [16], which applies the Sinkhorn-Knopp algorithm directly despite that, in many cases, the number of rows and columns is not equal. Though they achieve good performance in practice, a rectangular matrix cannot simultaneously have each it's columns and each of its rows sum to 1. To solve this, we employ a simple modification to the Sinkhorn-Knopp algorithm. Instead of normalizing *all* rows and columns, we normalize all rows and columns *except* the one row and column added as slack variables. This results in a matrix where all rows and all columns except the added row and column sum to one. In our experiments this achieves significantly better performance on both square and non-square matrices.

In addition we looked to compare against the the Dual Decomposition algorithm proposed by Torresani et al. [14]. We found, however, that running Dual Decomposition took several hours for a single image. Instead we turn to the next highest performing algorithm evaluated

by Torresani et al. [14]: Max-Product Belief Propagation (denoted "BP"). Here we transform the problem by removing the inequality constraints in Equation 3 and instead add them as a large cost term. Thus the function we maximize via Belief Propagation is as follows:

$$\max_{\mathbf{M}} S^{\text{bp}}(\mathbf{M}) = \sum_{a=1}^{n_1}\sum_{i=1}^{n_2}\sum_{b=a+1}^{n_1}\sum_{j=i+1}^{n_2} \mathbf{\Theta}^{\text{edge}}_{abij}\mathbf{M}_{ai}\mathbf{M}_{bj} + \alpha\sum_{a=1}^{n_1}\sum_{i=1}^{n_2}\mathbf{\Theta}^{\text{node}}_{ai}\mathbf{M}_{ai} - \eta\sum_{a=1}^{n_1}\sum_{i=1}^{n_2}\sum_{j=i+1}^{n_2}\mathbf{M}_{ai}\mathbf{M}_{aj} - \eta\sum_{a=1}^{n_1}\sum_{b=a+1}^{n_1}\sum_{i=1}^{n_2}\mathbf{M}_{ai}\mathbf{M}_{bi}, \quad (8)$$

$$\text{s.t. } \forall ai\, \mathbf{M}_{ai} \in \{0,1\},$$

where η is a large constant. As stated by Torresani et al. [14], this problem is equivalent to Equation 3, however this formulation introduces a large number terms to the energy minimization problem.

We also evaluate a Matlab implementation of our simplified Graduated Assignment algorithm against those algorithms compared in [27] on the CMU house image dataset. The results are shown in the Appendix.

A. Dataset

We evaluate our algorithms on two scenes from the TUM [44] dataset: the "fr2/large_no_loop" (which we denote fr2) and "fr3/long_office_household" (which we denote fr3) scenes. These scenes are indoor man-made structured environments, where we feel that wireframe reconstruction would be applicable. For each scene we sample two neighboring images from every 10 images. This results in a total of 335 test pairs for the first scene and 258 for the second. Notably, the rgb images in fr3 are rectified to have no lens distortion. The fr2 images, however, are not rectified but the distortion parameters are known. As the TUM datasets have ground truth trajectory information, we can compute the ground truth Fundamental Matrix and count the number of inliers whose Symmetric Epipolar Distance falls below a threshold. While this is not an exact measure of correct matches, it allows us to estimate the average improvement in the number of correct matches over the baseline.

VIII. Results and Discussion

For each algorithm the average percent improvement is calculated as: $\frac{1}{K}\sum_{k=1}^{K}\frac{\iota_k - \iota_k^{\#}}{\iota_k^{\#}}$ where K is the number of images in the scene, ι_k is the number of inliers found by the algorithm for image k, and $\iota_k^{\#}$ is the number of inliers found by the baseline algorithm. For reference we also present the non-normalized results calculated as $\frac{1}{K}\sum_{k=1}^{K}\iota_k - \iota_k^{\#}$. To remove scenes with heavy motion blur, or insufficient visual texture, we do not consider images where the nearest neighbor matching method fails to find at least 10 inliers.

The results for Wireframe matching are shown in Table I. For fr2, the nearest neighbor matching algorithm correctly matched an average of 44 nodes out of an average of 117 nodes detected in each image. For fr3, the nearest neighbor matching algorithm correctly matched an average of 62 nodes out of an average of 127 nodes detected in each image. We see that, in both scenes, sGA has near identical performance to Graduated Assignment, increasing the number of correctly matched features by over 25% when using only node similarity and over 30% when using both node and edge similarity. We see that the BP algorithm takes significantly longer than both sGA and Graduated Assignment and only achieves a 20-25% improvement, even when using both node and edge similarity. The increased runtime is likely due to the large number of terms added by transforming the uniqueness constraints.

fr2 Wireframe Matching Results

Algorithm	% Improv.	Avg. Improv.	Avg. FPS
sGA_Node	32.74%	12.60	267.52
sGA_Both	39.26%	15.24	259.74
GA_Node	34.74%	13.46	3.26
GA_Both	39.50%	15.32	3.34
BP_Node	8.84%	3.50	0.15
BP_Both	25.96%	10.39	0.15

fr3 Wireframe Matching Results

Algorithm	% Improv.	Avg. Improv.	Avg. FPS
sGA_Node	26.62%	14.77	235.95
sGA_Both	31.70%	17.43	228.63
GA_Node	28.12%	15.64	2.81
GA_Both	31.91%	17.61	2.89
BP_Node	6.28%	3.85	0.13
BP_Both	20.33%	11.75	0.13

TABLE I
WIREFRAME MATCHING RESULTS IN TWO SCENES FROM THE TUM DATASET

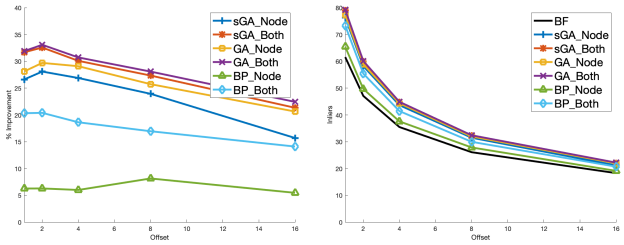

Fig. 3. Effect of increasing the time between images on fr3. Offset indicates the number of images between the compared images in the sequence. We see that the total number of inliers (right) decreases, but the percent improvement over the baseline (left) is maintained.

We further tested to see if these results would hold as the rotation and translation between the images increased. For this we performed a similar sampling as described in section VII-A except, instead of neighboring images, we chose images at an increased offset apart. As shown in Figure 3, the percent improvement does not change significantly as the offset is increased. This hints that, while our features' similarity matrix becomes less accurate as the offset increases, the wireframe extraction remains accurate, and therefore continues to enable improvement over the baseline. The small drop in accuracy by sGA_Node at the 16 image

offset may indicate a greater susceptibility of sGA to noise in the similarity matrices than Graduated Assignment.

The results for ORB feature matching are shown in Table II. For fr2, the nearest neighbor matching algorithm correctly matched an average of 132 nodes out of an average of 287 nodes detected in each image. For fr3, the nearest neighbor matching algorithm correctly matched an average of 175 nodes out of an average of 297 nodes detected in each image. In this experiment we see that sGA still achieves a 15% improvement, with the standard Graduated Assignment only giving a slightly better improvement. As the graph for ORB feature matching is somewhat less sparse than the wireframe case, this slight improvement may indicate that the additional iterations of Graduated Assignment are more important when there are more non-linear terms in the optimization problem. BP however fails to provide any benefit in this case. This could indicate a fragility to noise in the graph, or an inability to scale to large numbers of nodes.

fr2 ORB Matching Results

Algorithm	% Improv.	Avg. Improv.	Avg. FPS
sGA	23.05%	20.34	31.78
GA	24.08%	22.81	0.55
BP	1.05%	-4.95	0.01

fr3 ORB Matching Results

Algorithm	% Improv.	Avg. Improv.	Avg. FPS
sGA	15.51%	21.87	36.52
GA	17.49%	24.67	0.57
BP	-5.53%	-11.75	0.01

TABLE II

ORB MATCHING RESULTS IN TWO SCENES FROM THE TUM DATASET

IX. CONCLUSION AND FUTURE WORK

We have demonstrated a realtime algorithm for the matching of image wireframes that achieves significant improvement over nearest neighbor feature matching. We have further shown how this simplification of the Graduated Assignment algorithm can be applied to the standard feature matching problem. In the future we plan to look at how this algorithm could be applied to utilize other types of structure in a scene. For example, this algorithm could be used to match semantic object labels in sequences of images. Additionally we would like to look into utilizing a GPU for parallelization of both the standard Graduated Assignment algorithm as well as this simplified algorithm.

APPENDIX

Here we seek to justify the choice of the Graduated Assignment algorithm as the method for wireframe matching despite papers showing increased performance via other algorithms, and also encourage it's continued use and experimentation in the future. Specifically, many papers comparing against the Graduated Assignment algorithm appear to use the open-source version provided alongside [25]. While this implementation does not appear to be inaccurate, it does add an additional scaling step to the $n_1 n_1 \times n_2 n_2$ edge similarity matrix such that the full matrix sums to $n_1 n_1 n_2 n_2/2$. While normalization is generally a good idea, this results in a edge similarity matrix that is significantly different than the matrix described in [16] and requires different parameters to achieve good performance. As shown in Figure 4, for the experiments run in [27], the Graduated Assignment algorithm gains a significant boost in performance when removing this scaling step.

The algorithms compared here are Graduated Assignment (GA) [16], Probabilistic Matching (PM) [45], Spectral Matching (SM) [46], Spectral Matching with Affine Constraints (SMAC) [25], Integer Projected Fixed Point Method both Undirected (IPFP-U) and Directed (IPFP-D) [47], Re-Weighted Random Walk Matching (RRWM) [48], and Factorized Graph Matching both Undirected (FGM-U) and Directed (FGM-D) [27]. We also include a matlab implementation of our simplified Graduated Assignment algorithm (sGA) in these results. We see that for this task sGA has similar performance to the standard Graduated Assignment algorithm when the graphs are identical, and has a small ($\sim 5\%$) drop in accuracy when nodes have been randomly removed. Note that in this experiment we leave β_0 set to 0.5 as we are starting from a uniform initialization.

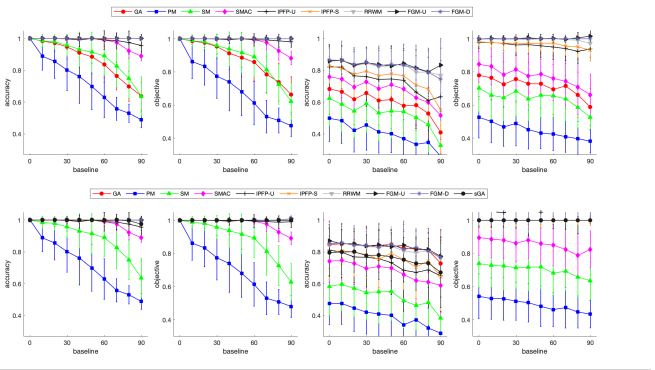

Fig. 4. Effect of removing the scaling on the edge similarity matrix. (Top) Original results from [27] applying graph matching to the CMU House sequence. (Bottom) Results without scaling. The graphs on the left are evaluated on identical graphs, while the ones on the right are evaluated on graphs with random nodes removed. In addition we include an implementation of sGA in these results.

REFERENCES

[1] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 1253–1260.

[2] O. Haines and A. Calway, "Estimating planar structure in single images by learning from examples." in *ICPRAM (2)*, 2012, pp. 289–294.

[3] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz, "Planercnn: 3d plane detection and reconstruction from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4450–4459.

[4] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, "Planenet: Piece-wise planar reconstruction from a single rgb image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2579–2588.

[5] Y. Zhou, H. Qi, Y. Zhai, Q. Sun, Z. Chen, L.-Y. Wei, and Y. Ma, "Learning to reconstruct 3d manhattan wireframes from a single image," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7698–7707.

[6] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[7] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Advances in neural information processing systems*, 2006, pp. 1161–1168.

[8] A. Saxena, S. H. Chung, and A. Y. Ng, "3-d depth reconstruction from a single still image," *International journal of computer vision*, vol. 76, no. 1, pp. 53–69, 2008.

[9] T. v. Dijk and G. d. Croon, "How do neural networks see depth in single images?" in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2183–2191.

[10] L. He, G. Wang, and Z. Hu, "Learning depth from single images with deep neural network embedding focal length," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4676–4689, 2018.

[11] D. Sharvit, J. Chan, H. Tek, and B. B. Kimia, "Symmetry-based indexing of image databases," in *Proceedings. IEEE Workshop on Content-Based Access of Image and Video Libraries (Cat. No. 98EX173)*. IEEE, 1998, pp. 56–62.

[12] C. Di Ruberto, "Recognition of shapes by attributed skeletal graphs," *Pattern Recognition*, vol. 37, no. 1, pp. 21–31, 2004.

[13] J. Liu, W. Liu, and C. Wu, "Objects similarity measurement based on skeleton tree descriptor matching," in *2007 10th IEEE International Conference on Computer-Aided Design and Computer Graphics*. IEEE, 2007, pp. 96–101.

[14] L. Torresani, V. Kolmogorov, and C. Rother, "Feature correspondence via graph matching: Models and global optimization," in *European conference on computer vision*. Springer, 2008, pp. 596–609.

[15] Y. Zhou, H. Qi, and Y. Ma, "End-to-end wireframe parsing," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 962–971.

[16] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 4, pp. 377–388, 1996.

[17] K. Huang, Y. Wang, Z. Zhou, T. Ding, S. Gao, and Y. Ma, "Learning to parse wireframes in images of man-made environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 626–635.

[18] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 4, pp. 722–732, 2008.

[19] S. Ramalingam, J. K. Pillai, A. Jain, and Y. Taguchi, "Manhattan junction catalogue for spatial reasoning of indoor scenes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3065–3072.

[20] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[21] S. Jouili and S. Tabbone, "Graph matching based on node signatures," in *International Workshop on Graph-Based Representations in Pattern Recognition*. Springer, 2009, pp. 154–163.

[22] I. S. Abuhaiba, "Offline signature verification using graph matching," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 15, no. 1, pp. 89–104, 2007.

[23] S. Jouili, I. Mili, and S. Tabbone, "Attributed graph matching using local descriptions," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2009, pp. 89–99.

[24] S. Serratosa, "Speeding up fast bipartite graph matching through a new cost matrix. int," *Journal of Pattern Recognition*, vol. 29, no. 2, 2015.

[25] T. Cour, P. Srinivasan, and J. Shi, "Balanced graph matching," in *Advances in Neural Information Processing Systems*, 2007, pp. 313–320.

[26] F. Zhou and F. De la Torre, "Factorized graph matching," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 127–134.

[27] F. Zhou and F. De la Torre, "Deformable graph matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 2922–2929.

[28] D. K. Lê-Huu and N. Paragios, "Alternating direction graph matching," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4914–4922.

[29] Z.-Y. Liu and H. Qiao, "Gnccp—graduated nonconvexityand concavity procedure," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 6, pp. 1258–1267, 2013.

[30] H. Almohamad and S. O. Duffuaa, "A linear programming approach for the weighted graph matching problem," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 5, pp. 522–525, 1993.

[31] A. Aboudib, V. Gripon, and G. Coppin, "A neural network model for solving the feature correspondence problem," in *International Conference on Artificial Neural Networks*. Springer, 2016, pp. 439–446.

[32] A. Zanfir and C. Sminchisescu, "Deep learning of graph matching," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2684–2693.

[33] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792.

[34] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2548–2555.

[35] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[36] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[37] L. Zhang and R. Koch, "An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency," *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 794–805, 2013.

[38] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[39] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 508–515.

[40] V. Kolmogorov and R. Zabih, "Multi-camera scene reconstruction via graph cuts," in *European conference on computer vision*. Springer, 2002, pp. 82–96.

[41] V. Kolmogorov and R. Zabin, "What energy functions can be minimized via graph cuts?" *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 2, pp. 147–159, 2004.

[42] A. Rangarajan and R. Chellappa, "Generalized graduated nonconvexity algorithm for maximum a posteriori image estimation," in *[1990] Proceedings. 10th International Conference on Pattern Recognition*, vol. 2. IEEE, 1990, pp. 127–133.

[43] R. Sinkhorn, "A relationship between arbitrary positive matrices and doubly stochastic matrices," *The annals of mathematical statistics*, vol. 35, no. 2, pp. 876–879, 1964.

[44] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[45] R. Zass and A. Shashua, "Probabilistic graph and hypergraph matching," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[46] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2. IEEE, 2005, pp. 1482–1489.

[47] M. Leordeanu, M. Hebert, and R. Sukthankar, "An integer projected fixed point method for graph matching and map inference," in *Advances in neural information processing systems*, 2009, pp. 1114–1122.

[48] M. Cho, J. Lee, and K. M. Lee, "Reweighted random walks for graph matching," in *European conference on Computer vision*. Springer, 2010, pp. 492–505.