Classification of Objects using Graph Spectra Embeddings derived from Hough Space

Duncan Mazza
Data Structures and Algorithms
Olin College of Engineering

Alexander Miller
Data Structures and Algorithms
Olin College of Engineering

***Abstract*—We propose an algorithm to quickly and accurately classify edge-detected images of isolated objects. This is motivated by the general problem of matching objects observed by cameras to corresponding 3D models' wire-frame renderings. Our algorithm uses a low-dimensional embedding of the edge-detected image that is a histogram-representation of the spectra of the graph created using the relationship of peaks in the image's Hough transformation. We narrow the problem scope to matching edge-detected images to rotated and translated instances of themselves, and see success rates of up to 90% in classification. We employ both linear search and simulated annealing optimization methods and compare their results.**

I. Introduction

The problem of matching a 3D model to its corresponding instance in the real world is a well explored problem. Often, it is framed as camera pose identification with respect to a marker-less object, where the object has a known geometry. Finding the relative pose often requires an optimization method that utilizes both an edge-detected image and the wire-frame or silhouette rendering of the 3D model and employs optimization techniques such as gradient descent with the objective function being some metric based on the Chamfer distance [12][9][5][10]. However, to determine which 3D model to look for in a scene without user input requires either an advanced machine learning algorithm or a fairly exhaustive template search.

In exploration of this problem of identifying an instance of an object in an image, we consider the narrower but related problem of classifying edge-detected images with their identical but rotationally and translationally transformed counterparts. Our process is compartmentalized into two main parts: creating an embedding of the object training images (§II - §III) and comparing those embeddings to the object test images at runtime (§IV-§V).

II. Embeddings the Hough Plane

A. Canny Edge Detection

The first processing step we applied was edge detection so that the visual structure of the object would be extracted. We chose the Canny edge detection algorithm in which the gradient for each pixel is calculated in both horizontal and vertical directions. After filtering with non-maximum suppression, the gradients are filtered by a threshold value; we empirically determined that a threshold value of 0.4 worked well for our purposes as it maintaining enough detail to preserve the shape of the object while getting rid of noise such as lettering on the object that could cause over-fitting.

B. Hough Transform

To extract a lower fidelity representation of the edge-detected image, we applied the Hough transform. In the Hough space, lines are represented as points on a plane [1] with axes of angle θ and distance ρ with respect to the origin, such that $x\cos(\theta) + y\sin(\theta) = \rho$. When the Hough transform is applied to an edge detected image, a line is rotated around each edge point of the image, and each of these lines represents a point in the Hough space. Based on the number of other edge points this line intersects, the corresponding point in the Hough space is given a weight. This means that long lines correlate with higher weighted points (also referred to as peaks) in the Hough domain. We empirically determined that preserving the 50 strongest peaks was a sufficient threshold.

III. Graph Generation

With the observation that the relative and not absolute position of peaks in the Hough plane encodes information about the edge-detected image, we treated the peaks as nodes in a graph. Using the Manhattan distance metric, we made an un-directed connection between nodes if the distance between them was below a certain threshold value. However, using this distance metric without any further adjustments reveals the problem that peaks with a θ value of $\approx -90°$ have a large Manhattan distance to peaks with a θ value of $\approx 90°$ and a rho of an opposite sign, despite representing similarly positioned and oriented lines. In order to take this into account, we shifted the peaks by $-90°$ and remapped points with θ values $< 90°$ into the 0 to $90°$ range and flipped the sign of their ρ values. After calculating the Manhattan distances between all the points for both the shifted and non-shifted transform, we retained only the minimum distances between the distances calculated for the original and shifted Hough peaks. A final step of filtering the distances under an empirically determined threshold was then applied, resulting in a graph with Hough peaks whose neighborhoods in the graph represent similarly oriented and positioned lines.

IV. Graph Similarity Comparison

Our classification algorithm is predicated on the assumption that the topology of the graphs is an embedding that contains

useful information about the relationship between line segments in the edge images. Operating with this assumption, we sought to implement an algorithm that would quantify with a scalar value the similarity between any two arbitrary graphs.

A. A Graph Spectral Theory-Informed Similarity Measure

We explored multiple options for comparing the similarity of graphs. A framing of the graph matching problem we explored is to perform an approximation of the un-directed graph isomorphism (or homeomorphism) problem, Two graphs are isomorphic if they share the same number of edges and nodes, and those edges and nodes are connected in the same way [8][1]. However, noise from the camera and edge detection in addition to the variations in the graph embeddings for identical but rotated and translated images prevent applicability determining graph isomorphism. Instead, we seek to use a different method of comparison that, like checking for isomorphism, considers the topology of the graph while being robust to noise and variations.

One approach we considered was an approach based in graph algorithms specifically designed for information networks. Zaeger et. al. in [14] suggested an algorithm predicated on the hub and authority relationships and expanding upon the work in [6] in which edges and nodes were assigned a score and the Hungarian algorithm was used to compare scores between graphs. However, this approach relies on directed graphs. We decided to use an un-directed graph representation, which does not lend itself to this approach.

Rather, we used a graph spectra-based approach. A primary concern of graph spectral theory is eigendecomposition of the Laplacian matrix of a graph. The Laplacian matrix, L, of a graph $G = (V, E)$ whose degree matrix is D and adjacency matrix is A, is defined as $L = D - A$. The normalized Laplacian matrix, $\mathcal{L}$ is defined as $\mathcal{L} = D^{-0.5}AD^{-0.5}$ with the convention that $D(v, v)^{-1} = 0$ [3]. Many qualities of a graph can be observed in the distribution of its Laplacian matrix's eigenvalues; as described in §1.3 of [3]: "Roughly speaking, half of the main problems in spectral theory lie in deriving bounds on the distribution of eigenvalues."

Our choice to use graph spectra was inspired by Shokoufandeh et. al.'s embeddings of directed acyclic graphs (DAGs) in [11] in which a topological signature vector (TSV) was created from the eigenvalue sums at the root of a DAG. Althogh these TSVs could be used to compare graphs, Shokoufandeh et. al. went further and suggested an approach involving signatures at each node's respective local subgraph in the DAG. Because the algorithm described in [11] attempted to solve a problem identical to the problem we set out to solve by creating graph representations of images, Shokoufandeh et. al. made this design decision to enable more robust object identification in the face of partial occlusions of objects and background clutter. For simplicity, we chose to make the assumption that the objects we would be identifying would neither have background clutter or be partially occluded.

As such, we employed a more simplistic method of graph spectra-based embedding. As described in [3], the set of eigenvalues of $\mathcal{L}$, $\mathcal{E}$, exist on the range $[0, 2]$. There are many nuances to the bounds of these eigenvalues, but this is true for any graph and is a key characteristic that our algorithm operates with. Following the approach in [4], we utilize this constraint to build histograms for each graph's eigenvalues and use the Wasserstein[2] algorithm to compare the histograms. A departure from [4] is that we consider histograms of eigenvalues instead of spectral measures which are probability density measures composed of Dirac measures at each eigenvalue (with normalized areas). This design choice was made to simplify our algorithm and is consistent with the use of probability density functions because the Wasserstein distance can be used in an analogous way with histograms of equivalent area.

A visualization of the core components of our algorithm described thus far can be seen in fig. 1.

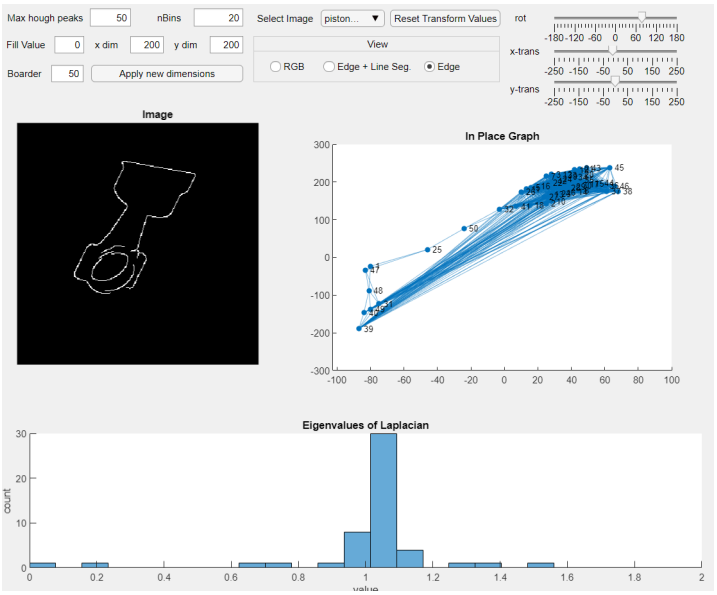

Fig. 1: Depicted is a screen capture of the interactive app we made to visualize the edge detection, graph embedding, and graph spectra components of our algorithm in the top left, top right, and bottom halves of the window respectively. The controls at the top right allow the user to transform the source image in real time, with the controls on the top left controlling the size and background values of the resulting image as well as the parameters for making the histogram.

V. Search Methods

We classify an image and its corresponding graph G with the class of the image in our data set whose graph, G', has

[1]Formally, two graphs $G = (V, E)$ and $G' = (V', E')$ are isomorphic if there is bijective mapping $f : V \to V'$ s.t. $\{f(x), f(y)\} \in E' \iff \{x, y\} \in E \ \forall \{x, y\}$ [13],[11].

[2]To expedite our implementation and due to a lack of a built-in implementation of the Wasserstein distance in MATLAB, we used open-source code from [7]. This code exists in our repository as the function `ws_distance.m`.

an eigenvalue histogram that is the smallest distance from G's eigenvalue histogram among all other graphs in the data set. We implement both linear search and simulated annealing optimization and compare their results.

For each image containing a distinct object, the image is rotated and translated with a user-defined set of angles and translation values. An example of how images are rotated and translated set is shown in fig. 2. The resulting set of images is distributed into two sets: one set for creating the database of eigenvalue histograms, and the other set for comparing against the database[3]. Performing linear search for a single image in the comparison data set, I, then, requires computing the Wasserstein distance between its eigenvalue histogram and each in the database.

Fig. 2: A sampling of the data set. The top left contains the original image, and the rest of the images are examples of rotated and translated versions of the original image which are randomly assigned to either the training or classification data sets.

The simulated annealing optimization, by contrast, does not calculate the Wasserstein distance between the image of interest and every image in the data base. Rather, there is an upper bound of iterations that the simulated annealing algorithm runs for, with the Wasserstein distance being calculated once each iteration. The simulated annealing keeps track of a variable, x, which represents the current eigenvalue histogram in the database that is being considered as a candidate for the classification of I. To generate each subsequent image histogram to compare against I's eigenvalue histogram, h_I, a heuristic is used that is part stochastic, part deterministic. Each time x is updated, the heuristic always first suggests another eigenvalue histogram that has the same classification label as x. If x is not updated in a particular iteration of the algorithm (following the updating convention of simulated annealing[4]), then the heuristic returns another x that is from a different classification; the heuristic rotates through each classification (the order of which is randomized each time x is updated), and never returns a particular eigenvalue histogram more than once.

[3]This database is simply implemented as a matrix of values in MATLAB.
[4]As described in [2], for example.

VI. Results

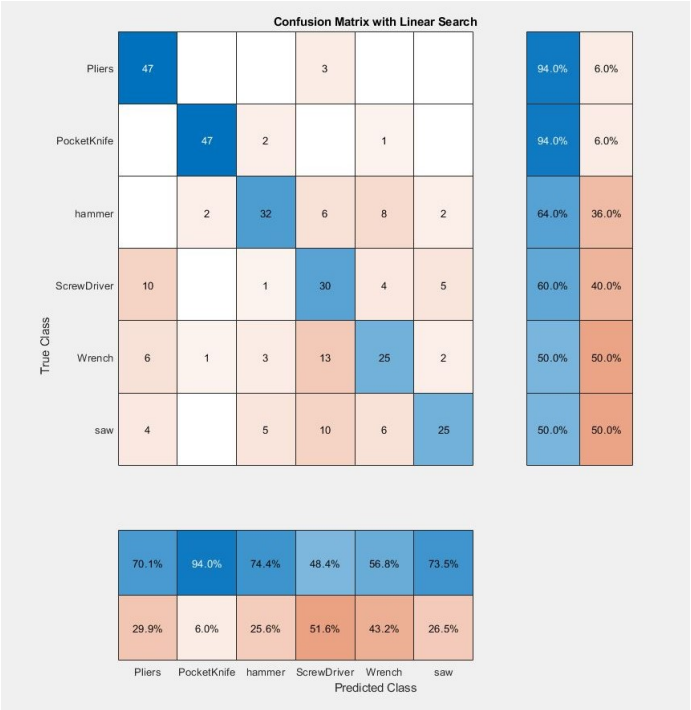

Fig. 3: Linear search results.

The reasoning behind this approach is that the prioritization of comparing eigenvalue histograms from the same class as x will ideally help the algorithm converge on a correct solution if x is of the correct classification. Further analysis of the behavior of the algorithm is needed to determine whether this behavior is exhibited or not.

Our algorithm was able to identify objects with some accuracy. Using the linear search method, we achieved an accuracy of $\approx 70\%$. Using simulated annealing search resulted in a slightly lower accuracy on average. Results of both linear and simulated annealing search are demonstrated in the confusion matrices in figures 3 and 4 respectively. Although these results suggest room for improvement - both in the large discrepancy of classification accuracy between objects and in the upper bound of accuracy, we believe our results demonstrate that the embedding captures useful information about the objects and that the simulated annealing search can yield results comparable to linear search with better time complexity.

Further work on this algorithm could include creating graph embeddings that are more invariant to rotations and translations and revisiting the heuristic used in the simulated annealing search.

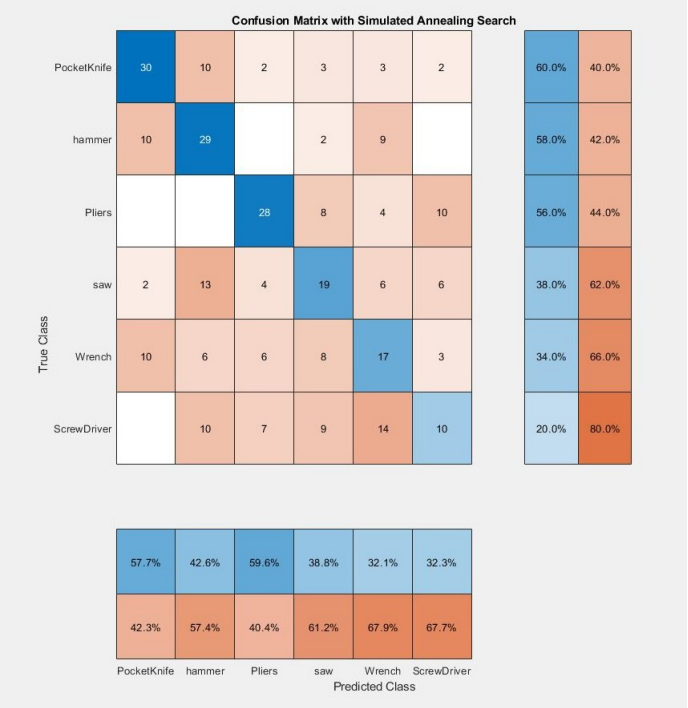

Fig. 4: Simulated annealing search results.

REFERENCES

[1] D. H. Ballard. "GENERALIZING THE HOUGH TRANSFORM TO DETECT ARBITRARY SHAPES*". In: *Journal of the ACM* 13.2 (1981), pp. 111–122. ISSN: 0031-3203. URL: https://www.cs.bgu.ac.il/~icbv161/wiki.files/Readings/1981-Ballard-Generalizing_the_Hough_Transform_to_Detect_Arbitrary_Shapes.pdf.

[2] Roger Carr. *Simulated Annealing*. URL: https://mathworld.wolfram.com/SimulatedAnnealing.html.

[3] F.R.K. Chung et al. *Spectral Graph Theory*. CBMS Regional Conference Series. Conference Board of the mathematical sciences, 1997. ISBN: 9780821803158. URL: https://books.google.com/books?id=4IK8DgAAQBAJ.

[4] Jiao Gu, Bobo Hua, and Shiping Liu. "Spectral distances on graphs". In: *Discrete Applied Mathematics* 190-191 (2015), pp. 56–74. ISSN: 0166-218X. DOI: https://doi.org/10.1016/j.dam.2015.04.011. URL: http://www.sciencedirect.com/science/article/pii/S0166218X15001912.

[5] S. Hinterstoisser et al. "Gradient Response Maps for Real-Time Detection of Textureless Objects." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence, Pattern Analysis and Machine Intelligence, IEEE Transactions on, IEEE Trans. Pattern Anal. Mach. Intell* 34.5 (2012), pp. 876–888. ISSN: 0162-8828. URL: http://olin.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.6042881&site=eds-live.

[6] Jon M. Kleinberg. "Authoritative Sources in a Hyperlinked Environment." In: *Journal of the ACM* 46.5 (1999), p. 604. ISSN: 00045411. URL: http://olin.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=2843783&site=eds-live.

[7] Nikalas Kolbe. *Github*. URL: https://github.com/nklb/wasserstein-distance.

[8] Raghvendra Kumar and Prasant Kumar Pattnaik. *Graph Theory*. Laxmi Publications Pvt Ltd, 2018. ISBN: 9789386035707. URL: http://olin.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2228702&site=eds-live.

[9] Ming-Yu Liu et al. "Fast object localization and pose estimation in heavy clutter for robotic bin picking". In: *The International Journal of Robotics Research* 31.8 (2012), pp. 951–973. DOI: 10.1177/0278364911436018. eprint: https://doi.org/10.1177/0278364911436018. URL: https://doi.org/10.1177/0278364911436018.

[10] Imperoli Marco and Pretto Alberto. "D2CO: Fast and Robust Registration of 3D Textureless Objects Using the Directional Chamfer Distance." In: *Computer Vision Systems : 10th International Conference, ICVS 2015, Copenhagen, Denmark, July 6-9, 2015, Proceedings* (2015), p. 316. ISSN: 978-3-319-20904-3. URL: http://olin.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edssjb&AN=edssjb.978.3.319.20904.3.29&site=eds-live.

[11] Ali Shokoufandeh and Sven Dickinson. "Graph-Theoretical Methods in Computer Vision". In: *Theoretical Aspects of Computer Science: Advanced Lectures*. Berlin, Heidelberg: Springer-Verlag, 2002, pp. 148–174. ISBN: 3540433287.

[12] Bin Wang, Fan Zhong, and Xueying Qin. "Robust edge-based 3D object tracking with direction-based pose validation." In: *Multimedia Tools and Applications* 9 (2019), p. 12307. ISSN: 1380-7501. URL: http://olin.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.585155026&site=eds-live.

[13] Eric W Weisstein. *Isomorphic Graphs*. URL: https://mathworld.wolfram.com/IsomorphicGraphs.html.

[14] Laura A. Zager and George C. Verghese. "Graph similarity scoring and matching." In: *Applied Mathematics Letters* 21.1 (2008), pp. 86–94. ISSN: 0893-9659. URL: http://olin.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0893965907001012&site=eds-live.