University of St Andrews

CS4099 Major Software Engineering Project

Graph matching with Lobsters

March 25, 2018

Supervisor:
Kasim Terzić

Submitted By:
Sizhe Yuen

Abstract

The ability to measure lobsters is important to be able to monitor the size and health of the creatures. By being able to do this task with images and automatically with software, we can aid scientists in this field accomplish their work more quickly. There is currently an existing dataset of images where efforts have been made to determine the size and sex of the lobsters using global features such as total length.

This project aims extend that work by representing images of lobsters as graphs and use graph matching techniques to compare between them. We aim to use these techniques to discover properties of the lobster, for example its age, gender and health. The effectiveness of these techniques will be evaluated against the existing dataset to discover if graph matching is a suitable method for lobster recognition and characterisation. Extensions to this project would be to develop a new algorithm for create graphs rather than using existing ones and to try the same techniques on more complex images with lobsters in their natural environment.

Declaration

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated.

The main text of this project report is NN,NNN* words long, including project specification and plan.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the report to be made available on the Web, for this work to be used in research within the University of St Andrews, and for any software to be released on an open source basis. I retain the copyright in this work, and ownership of any resulting intellectual property.

Contents

1 Introduction

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

1.1 Objectives

1.1.1 Primary objectives

- Explore and create suitable graph representations for lobsters
- Measure similarity of lobster graphs with existing software
- Automatically detect interest points from images
- Evaluate this method of graph matching on lobsters against the existing dataset

1.1.2 Secondary objectives

- Explore and create suitable graph representations for lobsters
- Measure similarity of lobster graphs with existing software
- Automatically detect interest points from images
- Evaluate this method of graph matching on lobsters against the existing dataset

2 Context Survey

2.1 Related work

2.1.1 Previous work

A large part of this project follows on from [1]. In his work, Abdallah created a dataset consisting of images and features of lobsters. The lobsters were measured and categorised and segmentation and feature extraction techniques were applied to create a more diverse dataset with baseline results. Additionally, classification and regression techniques were used to both classify the category of the lobster (juvenile or mature) and predict the carapace length.

This project is a continuation of Abdallah's work

2.1.2 Human pose recognition

The use of graphs in human pose estimation has been studied in the past [7] [13] where labelled nodes that represent important features such as hands and head are used to build skeleton models.

2.1.3 Cattle identification

2.2 Existing software

2.2.1 Graph visualisation

In order to visualise what a graph representation of a lobster may look like, graph drawing software that could import and export into a graph data format was needed. Initially the popular Graphviz and its `.dot` graph format [15] was explored. The `dot` graph format had all the attributes needed such as size of nodes and weights of edges, however there was no readily available GUI tool for drawing graphs as Graphviz mostly works on rendering existing `dot` files.

The open source Gephi [2] tool was exactly what was needed in terms of a graph drawing tools as it allowed a simple graph to be drawn with nodes and edges labelled.

2.2.2 Graph matching

Different graph matching and graph querying software was explored to deal with subgraph matching. What was needed was a tool that could find if a labelled subgraph was part of a larger graph in a database of pre-defined lobster graphs. The tool also had to be fast and able to query a large number ($> 100,000$) of subgraphs with sufficient speed. [6] [3]

3 Design

3.1 Annotation of dataset

The dataset provided by [1] was tagged with information on each image such as the lobster's sex, length of

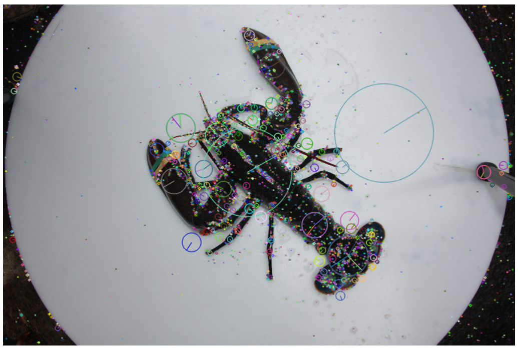
Figure 1: Example of annotated lobster image with nodes and edges of the graph perfectly matched.

3.2 Keypoint detection

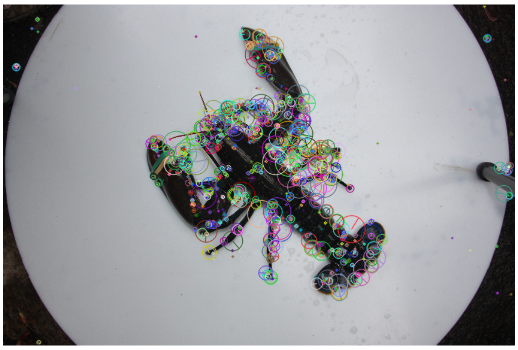
First, to identify important parts from our lobster images, keypoints or areas of interest must be identified. OpenCV [4] provides a host of different algorithms for feature detection such as Harris and Shi-Tomasi corner detectors and SIFT, SURF, ORB keypoint detectors [14]. All these algorithms were tried and tested on a small subset of the dataset to see if any would provide both useful and consistent features that can be used.

(a) Harris corner detection

(b) Shi-Tomasi corner detection



(c) SIFT keypoint detection

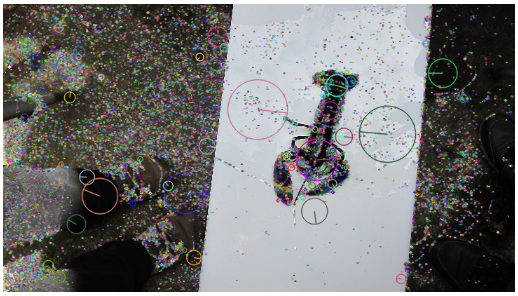


(d) SURF keypoint detection

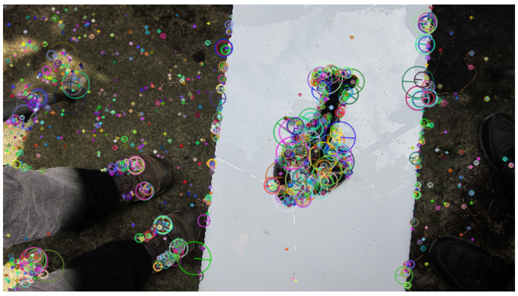
Figure 2: Comparison of different feature detection algorithms. The images have been scaled down after applying the detection to more clearly show the keypoints. Further comparison of the different detection algorithms on more images can be found in appendix A

(a) Harris corner detection

(b) Shi-Tomasi corner detection



(c) SIFT keypoint detection



(d) SURF keypoint detection

Figure 3: Comparison of different feature detection algorithms on an image with more noise.

From visually seeing the effects of each algorithm, it can be seen that the corner detectors do not work very well for our purpose. The shape of the lobster is not fully detected reliably. For images with a noisier background, much of the background such as slabs with sharp contrasting corners are also detected, as shown in figure 3. Furthermore, although corner detectors have applications in image matching [5], our goal from feature detection is to be able to extract graph like objects or features to apply graph matching on and so corner detection is unsuitable as we are not looking to match the different images to each other directly.

The keypoint detectors are able to provide better results for our objective, as we can see keypoints of important body parts being identified, such as the body, tail and claws. These keypoints are further able to be consistently identified from multiple images, showing that use of these algorithms are promising for automatically detecting the various parts of the lobster for construction of a graph. There is still a lot of noise in the detected keypoints, but we shall see later in section 3.3 that different methods can be applied to filter out unwanted keypoints. It is also non-trivial to create a graph from an outline of corners to represent the shape of a lobster, whereas the keypoints naturally translate well as nodes of a graph, with edges connecting them to form the shape and pose of a lobster. Because of these reasons, the keypoint detection algorithms in SIFT, SURF and ORB were further investigated while the corner detectors were discarded.

TODO fig of keypoint detector algorithm comparison

Figure 4: Comparison of different keypoint detection algorithms on multiple lobster images. See comparison of more images in appendix A

Between the different keypoint detection algorithms, SIFT was chosen as it gave the most consistent results and the kind of useful keypoints that are needed. TODO

3.3 Keypoint filtering

From just running a SIFT detector on the lobster images, it can be seen that there are a lot of small keypoints that are unimportant for our purposes. There are also many keypoints around the lobster that we would like to filter out, as we want all our keypoints to be on the lobster. Initially, the classic vision approach for feature matching using the keypoint descriptors [10] was tried, but the results obtained were surprisingly poor. Because of this, a more novel approach was taken for filtering. The small keypoints are filtered out by specifying an octave where all keypoints coming from that octave or above are kept. Finally, remaining keypoints that are not on the lobster are filtered with a colour histogram method where the colour histograms of the keypoints are compared and any below a certain difference threshold are filtered away.

3.3.1 SIFT descriptors

In computer vision, keypoint descriptors obtained from detectors like SIFT and SURF are often used for feature matching [9]. Lowe's paper [10] on the SIFT detector states that keypoints descriptors are highly distinctive, allowing a single feature to be correctly matching with good probability in a large database of features. This is exactly what we want, as we wish to extract the different features of a lobster (tail, claws, head). The only difference is we do not have a dataset of the lobsters, but not of dataset of individual lobster parts.

Because we do not have a dataset for individual parts, it made more sense to do the opposite of matching. Instead of using the distance between descriptors to match a detected keypoint with a known keypoint, the distance can be used as filter out keypoints that do not match closely to known ones. This distance can then be used as a threshold to filter more or less keypoints away.

As a test, the descriptor for the important body keypoint was taken from one image and calculated.

The descriptor was then matched to the closest other keypoints on another image to see if the body could be identified again.

TODO keypoint descriptor matching other image

Figure 5: The image on the left shows the keypoint that the descriptor was calculated from and the image on the right is the closest TODO matched keypoints to that descriptor.

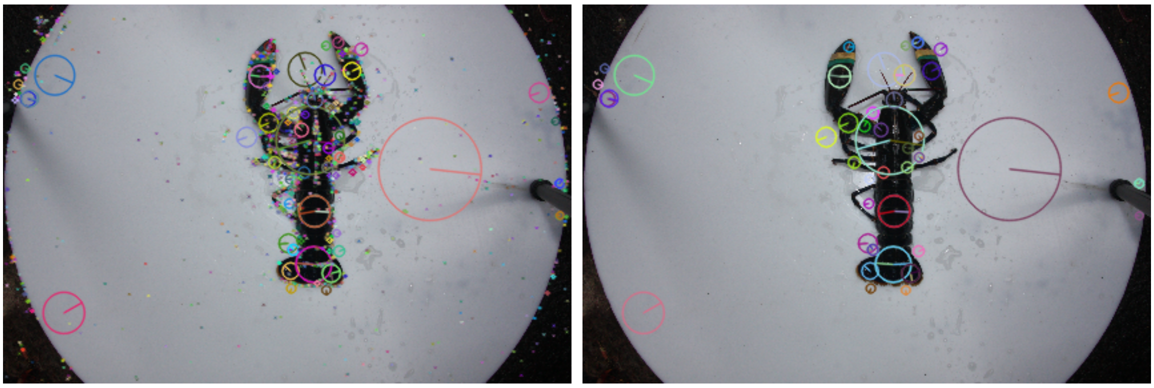
Figure 5 shows that the use of keypoint descriptors as a means of matching or filtering was not very reliable. TODO

As the traditional method of descriptors proved unreliable for our means, a slightly more novel method was needed to filter out keypoints. (TODO sentance wording)

3.3.2 Octave filtering

The method of filtering by the actual size of the keypoints was first looked at before looking at octaves. It was found to be less robust and less general than using octave levels. There are a few issues involved in using size of the keypoint for filter, namely how to choose a suitable threshold. The size threshold must be constant across all images, otherwise the method will not be able to generalise to unseen images. The size of the keypoints is directly related to the size of the original image, so any size filter threshold must be calculated based on the size of the image. This is not an issue, as a constant size threshold can be relative to the size of the image. However, with different sizes of lobsters, an aggressive threshold may remove important keypoints that we wanted to keep. Conversely a more conservative threshold would not remove enough keypoints and cause a large combinatorial explosion, a problem explained later in section 3.4.1 that we wish to avoid. This makes it difficult to set a good threshold as it would have to be arbitrarily defined and based solely on manual inspection of the images and keypoints sizes of the dataset. Furthermore, this seems to be quite a crude method TODO.

Octaves in SIFT are created by continually blurring an image. The idea behind this is to emulate looking at the image from different distances to get a varied set of keypoints. This means different features may be found at different octave levels. The high resolution of the images in our dataset causes many small keypoints to be found in the first few octave levels. These keypoints show many details that are irrelevant as we are concerned with the overall pose and size of the lobster.



(a) Before octave filtering

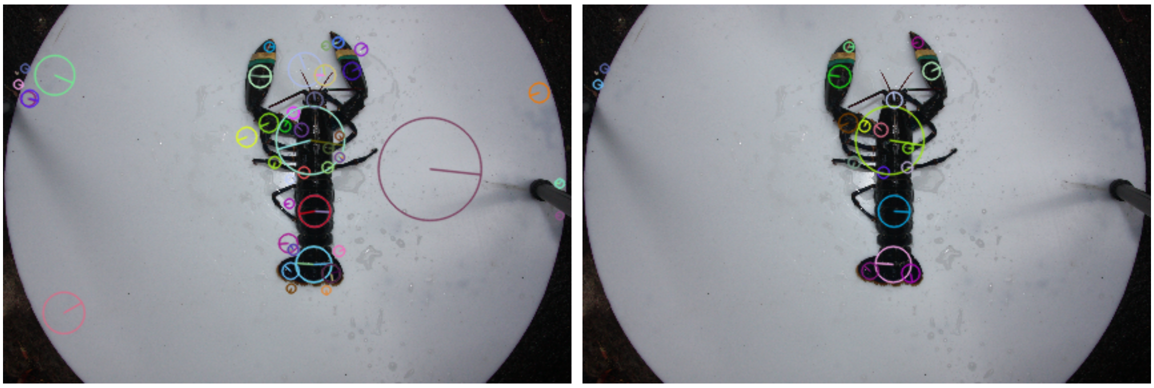
(b) After octave filtering

Figure 6: Before and after applying a filter on keypoints based on the octave level the keypoints were found in.

From this observation, we can apply a filter on all keypoints found below a certain octave level so that we are only left with the larger keypoints that capture the features we are looking for. A filter for all keypoints found below octave level 3 was used. This octave level threshold is highly dependent on the size and resolution of the original image. An image with lower size and resolution may need a lower threshold or none at all (TODO reason).

3.3.3 Colour histogram filtering

After applying the octave filter, there still remains some noisy keypoints we want to remove. Most notably are the keypoints found on the white background of the images. There have been studies [12] [8] that show applying a colour filter to eliminate unwanted feature points can be quite effective, especially if the background is very different from the target of the image. Following from this, colour histograms of each keypoints were calculated and compared to a set of pre-defined histograms. The difference between the histograms was compared and only keypoints whose different is above a certain threshold are kept.



(a) Octave filtering (b) Octave + colour histogram filtering

Figure 7: Difference between applying only octave filtering and applying both octave and colour histogram filtering.

The pre-defined histograms TODO

3.4 Graph creation

3.4.1 Probabilistic model

3.5 Graph matching

3.5.1 Subgraph matching

3.5.2 Subgraph rebuilding

4 Implementation

results why overlap classify [11]

5 Evaluation

6 Conclusion

References

[1] A. A. Abdallah. “Machine Learning with Lobsters”. MSc Thesis. University of St Andrews, 2017.

[2] M. Bastian, S. Heymann, and M. Jacomy. “Gephi: An Open Source Software for Exploring and Manipulating Networks”. In: (2009). URL: http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154.

[3] V. Bonnici et al. “APPAGATO: an APproximate PArallel and stochastic GrAph querying TOol for biological networks”. In: *Bioinformatics* Vol 32, Issue 14 (2016).

[4] G. Bradski. “The OpenCV Library”. In: *Dr. Dobb’s Journal of Software Tools* (2000).

[5] J. Chen et al. “The Comparison and Application of Corner Detection Algorithms”. In: *Journal of Multimedia* Vol. 4 Issue 6 (), pp. 435–441.

[6] R. Giugno and D. Shasha. “GraphGrep: A Fast and Universal Method for Querying Graphs”. In: *International Conference on Pattern Recognition* (2002).

[7] T. Hidenori, N. Atsushi, and T. Haruo. “Human Pose Estimation from Volume Data and Topological Graph Database”. In: *Computer Vision – ACCV 2007*. Springer Berlin Heidelberg, 2007, pp. 618–627.

[8] S. Hirai. “Color Filter in SIFT Matching”. In: *Proceedings of the 2013 JSME Conference on Robotics and Mechatronics*. Tsukuba, Japan, 2013.

[9] E. Karami, S. Prasad, and M. S. Shehata. “Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images”. In: *Computer Vision and Pattern Recognition* abs/1710.02726 (2015).

[10] D. G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *Int. J. Comput. Vision* 60.2 (Nov. 2004), pp. 91–110. ISSN: 0920-5691.

[11] K. Mikolajczyk et al. “A comparison of affine region detectors”. In: *International Journal of Computer Vision* Vol 65 (2005).

[12] C. F. Olson and S. Zhang. “Keypoint Recognition with Histograms of Normalized Colors”. In: *2016 13th Conference on Computer and Robot Vision (CRV)*. Victoria, BC, Canada: IEEE.

[13] M. Straka et al. “Skeletal Graph Based Human Pose Estimation in Real-Time”. In: *Proceedings of the British Machine Vision Conference*. 2011, pp. 69.1–69.12.

[14] opencv dev team. *Feature Detection and Description*. URL: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_table_of_contents_feature2d/py_table_of_contents_feature2d.html.

[15] *The DOT Language*. URL: https://www.graphviz.org/doc/info/lang.html.

Appendices

Appendix A Comparison of feature detection algorithms

TODO imgs

TODO more imgs