Project #4 Description
Regularity Discovery

CSE583/EE552 Pattern Recognition and Machine Learning, Spring 2023

Release Date: Monday, Mar 20, 2023
Submission Due Date: Monday, Apr 3, 2023

Contents

1 Introduction

In project 4, we will explore regularity discovery using PRML techniques. In this project, you will have the opportunity to investigate one of five different options for regularity discovery, including reflection symmetry, rotation symmetry, translation symmetry, recurring pattern, and reinforcement learning-based regularity discovery.
To begin, you will apply a baseline method to the provided dataset and evaluate its performance. You will then dive into recent research in their chosen area and explore new methods proposed in the literature. Finally, you will write a report summarizing your findings, observations, and thoughts on the broader topic of regularity discovery, with a particular focus on your chosen topic.

2 Regularity Discovery Topics

The following are six regularity discovery topics to choose from. Each topic has its own baseline method, dataset, and evaluation method, so please read through and select the one that interests you the most.

2.1 Reflection and Glide-Reflection Symmetry Detection

The goal of reflection symmetry detection aims to discover a reflection axis/plane that remains invariant under reflection.
Glide-reflection is another symmetry primitive, though less known about, that occurs frequently in the world around us. Glide-reflection is a combination of two symmetries. Similar to reflection, glide-reflection consists of an axis of reflection but also includes translation along that line. That is, glide-reflection is a symmetry composed of a translation along and a reflection about the same axis.

2.1.1 Dataset & Evaluation

We provide a 2D reflection dataset that includes 250 images with reflection axis labeled. Additionally, we provide a dataset of 40 images with glide-reflection labeled axes. The dataset consists of real-world images. Students can also explore other reflection datasets to gain a better understanding of the task. We offer an evaluation code that allows students to assess the performance of different reflection symmetry methods.

2.1.2 Methods to Investigate

We provide a baseline method that detects both reflection and rotation symmetry:

- Loy, Gareth, and Jan-Olof Eklundh. "Detecting symmetry and symmetric constellations of features." Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part II 9. Springer Berlin Heidelberg, 2006.

For glide-reflection, we offer the baseline:

- Lee, Seungky, and Yanxi, Liu. "Curved Glide-Reflection Symmetry Detection". Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence. 2011.

Students can use thes baseline methods as a starting point and explore recent research on reflection symmetry detection. For example, Seo et al. (CVPR 2022) proposed a method that uses equivariant learning for reflection and rotation symmetry detection. Students can also investigate other recent research papers to explore the latest advancements in the field.

- Seo, Ahyun, Byungjin Kim, Suha Kwak, and Minsu Cho. "Reflection and Rotation Symmetry Detection via Equivariant Learning." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9539-9548. 2022.

To our knowledge, the provided baseline is the most recent work on glide-reflection detection, though we encourage students to try reflection detection methods on the glide-reflection dataset.

2.2 Rotation Symmetry Detection

The goal of rotation symmetry detection is to find a rotation center invariant of rotation.

2.2.1 Dataset & Evaluation

We provide a 2D rotation dataset that includes 250 images with rotation centers labeled. The dataset is relatively challenging because all images are real-world examples. Students can also explore other rotation datasets to gain a better understanding of the task. We offer an evaluation code that allows students to assess the performance of different rotation symmetry methods.

2.2.2 Methods to Investigate

We provide a baseline method that detects both reflection and rotation symmetry:

- Loy, Gareth, and Jan-Olof Eklundh. "Detecting symmetry and symmetric constellations of features." Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part II 9. Springer Berlin Heidelberg, 2006.

Students can use this baseline method as a starting point and explore recent research on rotation symmetry detection. For example, Seo et al. (CVPR 2022) proposed a method that uses equivariant learning for reflection and rotation symmetry detection. Students can also investigate other recent research papers to explore the latest advancements in the field.

- Seo, Ahyun, Byungjin Kim, Suha Kwak, and Minsu Cho. "Reflection and Rotation Symmetry Detection via Equivariant Learning." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9539-9548. 2022.

2.3 Translation Symmetry Detection

Translation symmetry is one of the most commonly occurring symmetries in natural and man-made structures. Translation symmetry detection leads to the detection of lattice (tileable texture along 1D on the left, or 2D on the right in the figure below), which can be widely used in applications such as image de-fencing, texture editing, texture synthesis, and 3D reconstruction.
There are two types of translation (1) Translation with repetition in one direction, also known as Frieze Pattern. (2) Translation with repetition in two directions, also known as Wallpaper Pattern.

2.3.1 Dataset & Evaluation

To give students an opportunity to investigate translation symmetry detection, we provide a dataset of 170 annotated images that contain both Frieze and Wallpaper patterns. Translation instances in the dataset are labeled with polygons, making it easy to evaluate the performance of detection algorithms. Students can use this dataset to test and evaluate the baseline method and other approaches that they may investigate. We also provide an evaluation code for students to easily evaluate the performance among different translation symmetry methods.

2.3.2 Methods to Investigate

We provide two baseline methods for students to try out for translation symmetry detection:

- Leordeanu, Marius, et al. "Discovering texture regularity as a higher-order correspondence problem." (2006).
- Park, Minwoo, et al. "Translation-symmetry-based perceptual grouping with applications to urban scenes." Computer Vision–ACCV 2010: 10th Asian Conference on Computer Vision, Queenstown, New Zealand, November 8-12, 2010, Revised Selected Papers, Part III 10. Springer Berlin Heidelberg, 2011.

Students can use these methods as a starting point and compare their performance with other approaches. For example, Rodriguez-Pardo et al. (Computers Graphics 2019) proposed an automatic method for extracting and synthesizing regular repeatable patterns. Students are encouraged to explore other recent research papers on translation symmetry detection to gain insights into the latest advancements in the field.

- Rodriguez-Pardo, Carlos, et al. "Automatic extraction and synthesis of regular repeatable patterns." Computers & Graphics 83 (2019): 33-41.

2.4 Recurring Pattern Detection

Recurring pattern (RP) detection refers to the detection of "things that recur" in images, ranging from perfect symmetry to real-world similar objects. RP detection has numerous applications, including class-agnostic object counting, co-saliency detection, 3D scene understanding from a single image, and anomaly detection.
Notice: RP detection is a challenging task and the code requires student to have both Python and C++ coding backgrounds.

2.4.1 Dataset & Evaluation

We provide the RP-1K dataset containing approximately 1,000 images with RP ground truth. The dataset includes easy cases of images with perfect symmetry, as well as difficult cases of images with ambiguous RPs, even from a human perspective. Additionally, the dataset is split into easy and hard subsets, providing students the flexibility to evaluate RP detection methods on the entire dataset or just the easy subset. We offer an evaluation code for students to assess the performance of different methods.

2.4.2 Methods to Investigate

We provide a baseline method to detect RPs through local feature extraction and GRASP optimization.

- Liu, Jingchen, and Yanxi Liu. "Grasp recurring patterns from a single view." Proceedings of the IEEE conference on computer vision and pattern recognition. 2013.

Students can use this baseline method as a starting point for RP detection. While there are not many recent publications on RP detection, students are encouraged to investigate related topics such as unsupervised object detection. For example,

- Wang, Xudong, et al. "Cut and Learn for Unsupervised Object Detection and Instance Segmentation." arXiv preprint arXiv:2301.11320 (2023).

provides a code-available approach to this task.

2.5 Reinforcement Learning-Based Regularity Detection

Q-learning is a classical, model-free method for doing reinforcement learning. Q-learning aims to learn an optimal policy for an agent acting in an environment. In Q-learning, the agent learns a Q-function, which estimates the **expected** total reward that can be obtained by taking a particular action in a particular state and then following an optimal policy from that point on.
More recently, deep neural networks have been applied to and seen great success in RL. Deep Q-learning utilizes a neural network in value estimation for estimating future reward.

2.5.1 Environment and Evaluation

We provide students with an implementation of classical Q-learning as well as Deep Q-learning. Students will need to try and report on both methods.
We provide a maze environment where mazes are sized NxN. We utilize different map environments in the Q-learning and deep Q-learning programs, however both provide the ability to customize the map. We also provide an implementation to train the agent's policy. During training, the agent will execute episodes and gather reward. You will need to report on the agent's reward per episode and how quickly an optimal policy is learned for both methods.
Please note the RL code will instead be uploaded on a Github repo https://github.com/keatonkraiger/Q-Learning-For-Maze-Solving.

2.5.2 Your Investigation

We provide students with an implementation of the Q-learning algorithm for learning optimal policies in environments using classical and deep q-learning.
For Q-learning (classical approach), we implement the original algorithm:

- Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning. 1992.

For deep Q-learning, we follow the original that applied neural networks to Q-learning

- Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." Nature 518.7540. 2015

Here, instead of applying SOTA RL methods, we will be investigating the agent's performance on different maps and their configurations. Specifically,

- We provide different-sized NxN maps (with the ability to generate random ones) for the agent to train on. You should begin by training the agent on the different-sized maps and seeing how it performs.

- Once you have an idea of how well/poorly the agent does on different maps, you will be tasked with creating new maps (at least two for both classical and deep Q-learning). This map should have some form of regularity to it. Some examples include a repeating wall structure, passage, or entrance pattern (be creative!).

We encourage students to try different-sized maps and maze layouts. Note that you couldn't have a particularly regular pattern in a 3x3 maze so larger mazes may be necessary.

3 Grading Criteria

Depending on the topic you select, there will be different grading criteria. The options include any of the **symmetry detection or recurring pattern detection** in which you will try the baseline method as well as one other method. Option two is the RL-based method in which instead of trying a new or SOTA method, students will be applying the Q-learning algorithm on maze configurations of their own creation which contain some regularity to its design.

- All Topics: Investigation of the Baseline Method (**40 points**)
  - Summarize the baseline method provided in the project description. Provide a clear explanation of the method's key components and how it works. (10 points)
  - Apply the baseline method to the provided dataset/task and report evaluation results using the provided evaluation metrics or other commonly used metrics (with clear citation and statement). Results should be presented in either figure or table format. (20 points)
  - Describe your observations and reasoning about the baseline method's performance, including examples of images where it performed well and where it struggled. (10 points)
- Option One: Investigation of New Methods for Symmetry + Recurring Pattern Detection (**60 points**)
  - Clearly cite and summarize the new method(s) you investigated. Explain how the method(s) differs from the baseline method. (10 points)
  - Apply the new method(s) to the provided dataset and report evaluation results using the provided evaluation metrics or other commonly used metrics (with clear citation and statement). Results should be presented in either figure or table format. (20 points)
  - Describe your observations and reasoning about the new method(s) performance, including examples of images where it performed better or worse than the baseline method. Use the same images as in the baseline method part for comparison. (20 points)
  - Provide a clear and concise comparison of the baseline method and the new method(s), discussing the strengths and limitations of each method. (10 points)

  Option Two: Exploring Regularity in RL-based Learning (**60 points**)
  - While we don't try a SOTA method, find a new or SOTA method used in maze solving (or similar task such as grid-world based problems) to report on. Clearly cite and summarize this method and report on how it differs from classical q-learning. (10 points)
  - Describe your observations and reasoning about the agent's performance on the different maze configurations, including examples of images where it performed better or worse than the sample/randomly generate mazes. Do this for both the classical and deep Q-learning implementations. (20 points)

- Apply the Q-learning algorithm to the new mazes you've created (create at least two different maze configurations for either method) and report evaluation results using time to learn a policy as well as reward obtained per episode. Results should be presented in either figure or table format. Do this for both methods. (20 points)
- Provide a clear and concise comparison of the sample/randomly generated mazes to those you've created with regular elements, discussing the strengths and limitations of each maze configuration (if any). Further, provide some comparison between the two methods (deep and non-deep) in terms of speed, training, performance, etc. (10 points)

4 Submission

Your submission should include the following items:

- Option One: The output of each method/symmetry in the format the method natively outputs. Each output should be placed in an individual directory and made clear what method and dataset it belongs to.
- Option Two: The maps used for testing performance. This can either be in the form of a csv or numpy file. Be sure to label the maps.
- Your written report, which should be following the requirements & criteria listed above.
- A thorough ReadMe document that describes the various methods you ran with a link to their provided code.
- Please do not include the data file in your submission.

Note that including the data file in your submission is unnecessary and may cause issues with file size and upload time.
Please package all of these items into a single zip file and name it as

FirstName_LastName_ProjectNo.zip.

For example, **John_Doe_4.zip**.

Make sure to properly cite all resources you used for this project and double-check your submission before uploading it to the Canvas dropbox. Please note that graders will read and run your code, so please ensure that your code runs correctly and is well-organized and easy to understand.
Be sure to fully read the README.md for each of the provided baseline methods.

5 Common Issues

Notice that this section will be updated if more common issues are asked about by the students.