CP467: Image Processing & Recognition

Dr. Zia Ud Din
Group 1
Project Report

Fernando Garcia
Saif Al-Din Ali
Omer Shehab
Petar Mancic
Ozair Khan
Propa Roy
Isha Pabla
Jay Vora

Statement of Originality

All the code for this project has been written specifically for this course. The contents of this project does not contain any material that has been previously published or written by anyone that is not part of this group. Any aspect of this project that took inspiration from an online source has been cited below.

Citation from online sources: (ex. Task X took inspiration for Y from source Z:)

Statements of Contribution

Omer Shehab: Worked on the project report including literature reviews and methods/algorithms used in the code. Reviewed and tweaked around the code for task 1, 2 and 3 and conducted the necessary research on the methods used for the different tasks. Frequently communicated with the team and provided updates on progress.

Jay Vora: Created the initial version of the image dataset and was extremely involved throughout the creation process of subsequent datasets. Created the SIFT and BF matching implementation algorithms used in the early versions of task 1 and which were used in the feature matching process of task 3. Tested out different ML approaches and strategies for effectively fine-tuning the model. Frequently collaborated with Saif and the group for problem solving and issue identification.

Propa Roy: Worked on the project report and frequently communicated with the team and organized meetings. Researched alongside Isha for implementation details for how to fine-tune YOLO models and reviewed final implementation for task 2.

Isha Pabla: Worked on the project report and frequently communicated with the team and organized meetings. Researched alongside Propa implementation details for how to fine-tune YOLO models and reviewed final implementation for task 2.

Petar Mancic:

Ozair Khan: Helped Jay and Saif with the first version of task 1 using SIFT.

Fernando Garcia: Worked on code review, optimization, testing, and debugging. Also completed report sections on future improvements and challenges during implementation.

Saif Al-Din Ali: Created the final image dataset for tasks 1,2 and 3. Generated the augmented dataset used for fine-tuning the model through both local dataset creation and the online Roboflow data annotation and YOLO-compatible dataset. Created the final object identification and detection algorithm and custom bounding boxes. Implemented image stitching and panoramic image generation for task 3. Filled in the Results.xlsx and computed model accuracy and other statistics. Frequently collaborated with Jay and the group to trouble-shoot, problem-solve and implement different solutions. Did the final proof read of the report and edited the implementation method and challenges sections.

Task 1 Literature Review

Object detection is one of the most prominent domains in image processing techniques and has greatly been improved over the years. Techniques range from feature-based approaches to the new and modern deep learning frameworks, each with their own unique advantages and disadvantages.

1. Early Methods – Feature-Based Approaches

The foundation of object detection was initially built on the idea of recognizing visual patterns using pre-defined features. One of the earliest algorithms developed for this task was created by Voila and Jones back in 2001, named the **Haar Cascade** algorithm. This was one of the first forms of face detection, and it utilized Haar-like features and integral image representation to achieve this. This algorithm was fast, however, it was hindered by its dependency on rigid, predefined features.
Another technique was developed around 4 years later, by Dalal and Triggs in 2005, in which they created the **Histogram of Oriented Gradient**s (HOG). This method extracted gradient-based features that managed to capture the shape and structure of objects pretty well. Although this method was good at detecting pedestrians, it did not perform well with different poses and perspectives of the background.
Another notable method, **Scale-Invariant Feature Transform** (SIFT), was developed by David Lowe in 1999. This is one of the most influential techniques that excels at identifying and describing local features in images and making them robust for applications. The method identifies key points and applies orientation based on the local gradient direction and its vicinity. Keypoint descriptors are then generated by computing gradients on 4x4 subregions around the key points. This method is extremely robust to scale, rotation, and illumination changes, and is still often used in the modern day. However, like other older methods, it is computationally expensive and is easily outperformed in speed and accuracy by modern methods, such as CNN.

2. Modern Methods – Deep Learning Models

Convolutional Neural Networks revolutionized object detection in the modern era which turned away from handcrafted features, and focused on learning features from data. Many different versions have since surfaced most notably **AlexNet, VGGNet, ResNet.** AlexNet, proposed in 2012, made the breakthrough in deep learning for image recognition using ReLU activations, dropout regularization, and GPUs for training. VGGNet was known for its simplicity and introduced the concept of stacking small convolutional filters (3x3) to achieve deeper architectures. ResNet introduced residual connections which enabled the training of extremely deep networks. CNNs learn hierarchical features directly from data and raw inputs, which greatly decreases the need for manual labor. These models are also very flexible and scale effectively with larger and more complex datasets. However, the main limitations include being very data dependent, requiring large datasets for effective generalization. Additionally, CNN models can become very computationally expensive for training and inference.
Region-Based Convolutional Neural Networks was introduced shortly after, in 2014 by Girshik. This was a model which created ~2000 region proposals and applied CNNs to each proposal for feature extraction and classification. This model is very accurate, however, it is also very slow and demands intensive computational power.
Single Shot Detectors simplified object detection by eliminating the step for region proposal. **YOLO (**You Only Look Once) was introduced in 2016 by Redmon, and changed object detection into a regression problem. The model processed the entire image in a single forward pass, which was tremendously quick. However, this speed did not translate into accuracy, as this model struggled with identifying and detecting small objects, especially in a crowd.

Task 2 Literature Review

Bounding boxes are an integral part of object detection which serves as the spatial representation of an object's location in an image. The bounding box can be used to evaluate many of the object detecting algorithms. The advancement of object detection also greatly affected the advancement of bounding boxes simultaneously.

1. Early Methods

Similar to feature-based object detection, bounding-box object detection relied heavily on handcrafted features and sliding windows. The early methods include the **Haar Cascades** and **Histogram of Oriented Gradients (HoG)**. These were effective methods, and used a sliding window to scan over an image and apply a classifier to predict if an object was present or not. These approaches were really good for the early days and used in models for detecting facial features and pedestrians. However, like many old techniques, required a ton of computational power and were also inaccurate when it came to large clutters and crowded scenes.

2. Modern Methods

The more modern methods for creating and predicting bounding boxes were greatly revolutionized by deep learning breakthroughs, namely, the innovation of **Convolutional Neural Networks**. Models that expand on CNN, like the **Region-Based Convolutional Neural Network**, made it so bounding boxes were predicted as part of the object detection pipeline. This was done at the initial step where region proposals were first generated in the model, and then refined by applying CNNs to them. Faster R-CNN further expanded on this process by sharing features and replacing the region proposal step with region proposal networks which improved both the speed and accuracy of the original R-CNN model. Although these models were great at predicting bounding boxes, there still was difficulty in crowded scenes and irregular shapes, which were not ideal for rectangle bounding boxes.

Another advancement, much like for object detection, **Single-shot detectors** like YOLO and SDD (**single shot multibox detector**) removed the need for region proposals altogether. Instead opting to predict bounding boxes directly from the image itself in a single pass. This significantly improves the speed of the model, usually at the cost of accuracy. The **YOLO** framework treats object detection as a regression problem which predicts bounding boxes along with class labels from the image. This method is effective for quick, real-time application, however, suffers in precision when it comes to smaller objects and crowded scenes. Similarly to YOLO, **SDD** predicts bounding boxes for objects in an image in a single pass. However, this method shines better than YOLO when it comes to detecting small images. This is done by creating feature maps at different resolutions which greatly increases its ability to detect boxes at various scales.

Task 3 Literature Review

Image stitching is the process of combining multiple images to create a single, larger image. This technique, which involves multiple separate steps, is widely popular for creating scenic panoramas. Over the years, image stitching has evolved greatly from simple manual methods, to advanced algorithms and deep learning models.

1. Early Methods

Early methods of image stitching, similar to object detection and bounding boxes, were very dependent on handcrafted features and adopted a feature-based approach. The initial step of stitching

images is to first detect similar features in the images. The main method of these techniques was to identify key points or features in the images and detect overlapping regions and then align them using geometric transformations. The most influential technique was developed by David Lowe, namely the **Scale-Invariant Feature Transform** (SIFT). SIFT, when deployed for image stitching, would detect and describe invariant features in images. These were then used for matching corresponding key points across the different images. Later development multiple years later showcases **Speeded-Up Robust Features** (SURF), which was a faster alternative to SIFT. Both of these methods were ideal for feature extraction of the images while retaining robustness to rotation and illumination.
The second step to image stitching involves the geometric transformation, which involves trying to align the images. The most profound method involved **RANSAC** (**Random Sample Consensus**), which applied geometric transformation through homography. This involved mapping one image onto another by relating points in the images using a matrix, a homography matrix. RANSAC is best suited for dealing with outliers and noisy data from feature selection.
Lastly, the image needed to be seamlessly blended together for the final product. However, even with good feature selection and alignment, there still were problems due to lighting, exposure & position. Early techniques for blending include **feathering**, which blended pixel intensities at the boundaries. Additionally, **multi-band blending** used gaussian pyramids to blend images in frequency space. More advancements were made with the **Poisson image editing** which involved adjusting the gradient of the pixel intensities at the boundaries to improve seamless stitching. However, much of these early methods for image stitching suffered from occlusion handling and real-time processing.

2. Modern Methods

More modern advancements in deep learning have introduced **Convolutional Neural Networks**. This deep learning model combined the feature extraction and alignment process directly from the data. A new method to blend images, **Deep Image Blending** was introduced to seamlessly blend stitched images. This was done by automating the blending component using a neural network that was trained on large datasets. This approach was revolutionary, as it was much better at handling complex scenes with different lighting and perspective.
The most modern versions of image stitching involve multi-view, or 360-degree stitching, which involves stitching together images that have been captured from different viewpoints. This is often done using specialized cameras, however, the major issue with this is precision for image alignment. This technology is even being used for virtual reality headsets. This is done using panoramic projection techniques that map the images onto a spherical surface and then align them. The most notable systems designed for this are **PhotoSphere** and **AutoStitch**.

Implementation Methods and Rationale for Feature Detection

The initial method we employed for feature detection was SIFT because we knew from the outset that the different objects and images in the scene would have different perspectives and rotations as well as us being familiar with it from class. SIFT ensures that keypoints can be matched even if images are taken from different distances and angles which makes it the ideal method for feature detection. Additionally, SIFT is very good when it comes to capturing rich local information that helps distinguish between similar and different regions. However, when implementing the SIFT algorithm directly we found that it was very unreliable in its ability to detect objects. We first believed that there was too much noise in the scene throwing off the detection accuracy so we manually removed the background for all 30 scene images. This still didn't solve the issue and the model had to be very heavily tuned for simple identification. We experimented with various ML-based approaches and in

the end we found that transfer learning was the most reliable and valid method for feature detection that we tried.

Implementation Methods and Rationale for Feature Matching

In addition to SIFT, the initial method we utilized for feature matching was a Brute Force Matcher, whose main purpose is to match the features detected in one image, to those in another by comparing the feature descriptor. Similar to feature detection this method was inconsistent and required a lot of manual fine tuning rather than being robust to handle unfamiliar situations. Using transfer learning enabled us to effectively and reliably perform feature matching across most of the objects in the scene under a variety of scene conditions (i.e. the final dataset we used had a very noticeable glare from the lights overtop the table, this method was able to detect the objects in spite of these obstructions, whereas SIFT + BF was very sensitive to external noise and even when we removed the background it was still inconsistent).
To effectively implement feature matching through transfer learning it was imperative that we find a way to reliably train the model given our very limited image dataset. The method we used was a hybrid of data augmentation for a local dataset and data annotation through the online platform Roboflow. Roboflow is a platform that provides a lot of tools for computer vision and dataset creation. The main reason we chose it was because it allowed us to automatically turn our local dataset into a format compatible with YOLO. Another feature that Roboflow gave us is data annotation. We went manually through all 160 images we managed to generate from our small dataset and annotated each of them. This reduced the amount of noise the model picked up in training which was crucial for us as we didn't have a lot of data to train the model with. This data annotation allowed us to maximize the effectiveness of our model training to reliably detect and match the features in the scene.

Implementation Methods and Rationale for Image Stitching

We used RANSAC for homography estimation and Gaussian Window for blending and smoothing in order to stitch the images together. The main task of RANSAC is to compute a transformation matrix in order to align one image with another. This method works really well with SIFT and Brute Force Matching and so we decided it was best to start with this approach and it worked out really well. RANSAC handles perspective distortions very well which was a huge factor for us, since all our objects were captured at different angles and perspectives. The main task of Gaussian Window was to merge the aligned images into a single panorama with seamless overlapping regions. This method was ideal because it is very easy to implement and adjust by modifying the smoothing window size which made it easier to work with for our dataset. Additionally this method is very nice at blending and helps merge images with different brightness and texture smoothly, which make the panorama appear natural and flawless. RANSAC works amazingly well with SIFT and BFM and so it was the best choice for homography estimation. Gaussian smoothing was also the most ideal method for stitching the images seamlessly since we had images with different perspectives and lighting, and this method provided a clear and seamless result. To stitch together 10 images we simplified our algorithm to effectively stitch 2 images together. From there, we ran it iteratively starting with S1 and S2 and using the resulting image as the first pair of the next iteration. In the end, the image was adequately stitched together and if the images had sufficient overlap, little to no artifacts remained. We used these tutorials for the simple image stitcher: first, second and third.

Challenges Faced & Resolution Strategies

Feature Detection

Since most of the images in the dataset were very simple with low texture and repetitive patterns, it was hard to adjust the algorithms to properly detect these objects in the different scenes. Additionally, capturing the same objects at different perspectives and rotations involved more challenges when it came to creating the dataset and panorama.

We overcome these challenges through adjusting our approach. We initially started with a low difference in perspective in order to see how our model would handle it and then decided to make it more challenging by adding in different perspectives, illumination and rotation to the objects and then adjusting our code based on what it was lacking.

Task 1

Task 1 was our biggest hurdle for this project and required the most amount of work from members in order to work. We initially started by using SIFT for feature detection in order to detect the different objects and name them. It was especially hard getting the algorithm to detect the objects in a scene when the perspectives & rotations were altered which made it especially challenging. We tried removing the background from images to reduce the noise, but even so, it was proving difficult to detect objects in a scene accurately and so we ultimately changed our approach. We decided to instead use machine learning models in order to detect the objects and scenes. The main problem encountered with using pretrained models was identifying objects that didn't belong to classes it was trained on. To effectively perform transfer learning we either needed to rapidly source datasets for objects in classes that didn't belong to the model or select objects valid to that model and fine-tune specifically for our object (e.g. we trained our model on the specific book we used). The latter approach was far more successful in both speed of training and resulting accuracy.

The Gray Ball

The Gray ball wasn't able to be identified in most images due to its high similarity with the background color and its general shape which tends to reflect the color of the background onto its edges. This quality of the item made it difficult to detect and thus should of being omitted from the dataset but served as a prime example of something to improve upon

Possible Improvement & Future Work

By using shadow detection and alignment it might be possible to generate a bounding box around the silver ball, but seeing as this method would be very difficult to implement with the given time and concepts covered in this class we were not able to apply it for this project. In the future we would like to further study shadows and their relations to the object such that they may be used for proper item detection of items that either have poorly defined edges, or are omitted but their shadow is present. By using shadows and the physics of lighting we might be able to generate a shape of the object that is obscured and thus generate its bounding box. A study from MIT suggests that using light beams that generate a shadow to detect an object is possible and provides good results for computer vision applications, in our case these shadows are already generated by the lighting, but the same principles can be used.

Another future improvement is the creation of a more thorough and robust dataset for transfer learning. Given the restrictions in image number this project wasn;t able to maximize the potential of transfer learning. A more complete and robust model will invariably increase the accuracy, reliability and validity of the model.

Citations

A Comparative Study on Different Image Stitching Techniques: Jaya, P., & Anitha, S. (2021). A comparative study on different image stitching techniques. *International Journal of Engineering Trends and Technology*, 70(4), 205-209. Retrieved from https://ijettjournal.org/assets/Volume-70/Issue-4/IJETT-V70I4P205.pdf

Automatic Panoramic Image Stitching using Invariant Features: Brown, M., & Lowe, D. G. (2007). Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1), 59-73. Retrieved from http://www2.ene.unb.br/mylene/PI/refs/ijcv2007.pdf

Convolutional Neural Network Series 5 — The CNN Hall of Fame: Exploring Classic Convolutional Network Architectures: Zhang, R. (2021). Convolutional Neural Network Series 5 — The CNN Hall of Fame: Exploring Classic Convolutional Network Architectures. Retrieved from https://rendazhang.medium.com/convolutional-neural-network-series-5-the-cnn-hall-of-fame-exploring-classic-convolutional-f42712f84002

Convolutional Neural Networks: A Brief History of their Evolution: AppyHigh. (2021). Convolutional Neural Networks: A Brief History of their Evolution. Retrieved from https://medium.com/appyhigh-technology-blog/convolutional-neural-networks-a-brief-history-of-their-evolution-ee3405568597

Deep Image Blending: Zhang, R., & Lalonde, J. F. (2020). Deep image blending. *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*. Retrieved from https://openaccess.thecvf.com/content_WACV_2020/papers/Zhang_Deep_Image_Blending_WACV_2020_paper.pdf

Distinctive Image Features From Scale-Invariant Keypoints: Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 91-110. Retrieved from https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

Focal Loss for Dense Object Detection: Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*. Retrieved from https://arxiv.org/pdf/1708.02002

Histograms of Oriented Gradients for Human Detection: Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Retrieved from https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

Image Processing Techniques: What Are Bounding Boxes?: Keymakr, (n.d.). Image Processing Techniques: What Are Bounding Boxes? Retrieved from https://keymakr.com/blog/what-are-bounding-boxes/

Image Stitching Techniques: Maponga, Y., & Tufts, D. (2017). Image stitching techniques. Retrieved from https://sites.tufts.edu/eeseniordesignhandbook/files/2017/05/Yellow_Maponga_F1.pdf

Object Recognition from Local Scale-Invariant Features: Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision (ICCV)*. Retrieved from https://www.cs.ubc.ca/~lowe/papers/iccv99.pdf

Poisson Image Editing: Perez, P., Gangnet, M., & Blake, A. (2003). Poisson image editing. *ACM Transactions on Graphics (TOG)*, 22(3), 313-318. Retrieved from https://www.cs.jhu.edu/~misha/Fall07/Papers/Perez03.pdf

Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography: Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), 381-395. Retrieved from https://www.cs.ait.ac.th/~mdailey/cvreadings/Fischler-RANSAC.pdf

Rapid Object Detection using a Boosted Cascade of Simple Features: Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Retrieved from https://pdfs.semanticscholar.org/presentation/12a3/76e621d690f3e94bce14cd03c2798a626a38.pdf

Shadow Item Detection: Somasundaram, S. (n.d.). Using shadows in space-time to see hidden objects. MIT Media Lab. Retrieved from https://www.media.mit.edu/projects/using-shadows-in-space-time-to-see-behind-occluders/overview/

SSD: Single Shot MultiBox Detector: Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot MultiBox Detector. *arXiv preprint arXiv:1512.02325*. Retrieved from https://arxiv.org/pdf/1512.02325

SURF: Speeded Up Robust Features: Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. *Proceedings of the European Conference on Computer Vision (ECCV)*. Retrieved from https://people.ee.ethz.ch/~surf/eccv06.pdf

The Evolution of Object Detection Methods: Jian, X., & Zhu, H. (2024). The evolution of object detection methods. *Engineering Applications of Artificial Intelligence*. Retrieved from https://www.sciencedirect.com/science/article/pii/S095219762400616X

The Viola-Jones Algorithm: Baeldung, (n.d.). The Viola-Jones Algorithm. Retrieved from https://www.baeldung.com/cs/viola-jones-algorithm

You Only Look Once: Unified, Real-Time Object Detection: Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*. Retrieved from https://arxiv.org/pdf/1506.02640