A historic image processing pipeline for measuring urban growth: A case study of Pyongyang

Authors:

Pari Sharma C078, Abhyudit Agarwal C077

Course: Image and Video Processing

Abstract

Tracking land-use changes resulting from processes like rapid urbanization is a vital assessment for urban development and environmental science. While some of the available modern deep learning methodologies and models are promising, they often require large amounts of labeled datasets, and computationally expensive models proven on large-scale replicable datasets. This paper presents a classic, interpretable pipeline to establish a framework for measuring high-contrast urban change. Using the analysis of images from space-based acquisitions of Pyongyang, DPRK approximately in 2014 and 2024, the method isolates and quantifies new construction, specifically a new apartment complex on Ryomyong Street. The pipeline included steps for image registration, conversion to grayscale, normalization via histogram equalization, image differencing to generate a change map, thresholding, and foundational morphological operations to clean the change mask. Our findings through this means, using this classical methodology, that the pipeline indicates an approximately 30.98% change within the selected area which corresponds to the isolated evaluation of new large-scale construction, where our process examined changes in construction to filter minor noise and minor seasonal or lighting changed. The entire workflow was developed and integrated into a user-friendly ascii tkinter application, that simply illustrates the point that quantitatively analysis of this nature through classic parameter driven methodology are a viable means of geographic analysis.

1. Introduction

The swift development of urban land, known as urban sprawl, challenges sustainable development. To manage resources, develop infrastructure, and determine environmental impacts, stakeholders (e.g., policymakers, urban planners, and environmental scientists) need appropriate and current data about land-use change. Satellite imagery can provide a comprehensive and large-scale data source for solving this problem. Traditionally, this data was

evaluated through an exhausting and costly manual approach. Lately, lots of this work has switched to an automated approach and the use of deep learning (DL) and Convolutional Neural Networks (CNN) can produce accurate results in image segmentation. However, many of these methods require vast amounts of hand-labeled training data, and are often criticized for their "black box" identification process. This paper explores a "middle ground": the classical, algorithm pipeline of computer vision. We hypothesize that when assessing high-contrast change events, like the construction of new buildings on previously undeveloped land, a well-tuned classical pipeline can yield results comparable to a DL model while being more transparent, cheaper, and easier to implement.

To evaluate this hypothesis, we established a multi-step workflow aimed at analyzing a decade of urban growth in a specific, rapidly developing area of Pyongyang. This paper will detail the methods, the iterative tuning, and results obtained, all through a custom desktop application we constructed.

2. Methodology

The core of our project is a sequential pipeline that processes a "before" and "after" image pair to develop a complete, quantifiable change mask. This process was carried out in Python, OpenCV, and Matplotlib.

2.1. Data Acquisition and Registration

The subject of this case study is the Ryomyong Street apartment complex in Pyongyang. The site was selected because it has undergone dramatic, contrasting transformation from 2014 (holding farmland and small built structures) to 2024 (holding many high-rise apartments).

- Before Image: raw_before_2014.jpg (c. 2014)
- After Image: raw_after_2024.jpg (c. 2024)

The images were sourced from Google Earth. An important step in the change detection process is image registration, ensuring both images are perfectly aligned. This was done in two phases:

Manual: During capture, the camera angle and zoom are kept the same.

Automated: The tkinter application programmatically resized both images to the lowest common dimensions–(which in our test case, was: (1302, 781)), making perfect use of 1-to-1 pixels.

2.2. The Processing Pipeline

The analysis process utilized in this example closely corresponds to the major steps reflected in the experiments of the "image processing" workflow.

1. Pre-processing: The two registered images are now in the 8-bit gray scale (logical type), if not already converted. We will change both images using a 5x5 Median Blur, to reduce "salt-and-pepper" noise from the satellite sensors while preserving important structural edges (graphically less) from getting blurred.
2. Normalization: There will be different shadows, lighting and atmospheric effects with satellite images a few years apart. So we apply Histogram Equalization to the two denoised, gray scale images, to normalize brightness and contrast. This ensures we can assess that a dark patch in the pre-satellite image is appropriately comparable to a dark patch in the post-image satellites.
3. Image Differencing: Next we utilize cv2.absdiff() to significantly subtract the normalized "before" image from the normalized "after" image and obtain the "difference" image (diff_image). The pixel values in the diff_image will be dark (black) if pixels are similar between the two images and light (white) pixel values if there has been considerable change.
4. Thresholding: Then we will threshold the diff image to create a binary thresh_mask, converting the diff_image from continuous values (grayscale) to binary pixel values. This is our first and raw effort to isolate "changed" pixels.
5. Morphological Refinement: The raw thresh_mask is very prone to noise due to minor changes from shadows changing or vegetation change! We will "clean" the mask with two different morphological operations:
    - cv2.MORPH_OPEN (Erosion followed by Dilation): This operation "opens" the image, breaking small connections and removing all the small white "salt" noise dots.
    - cv2.MORPH_CLOSE (Dilation followed by Erosion): This operation "closes" the image, filling in small black "pepper" holes in the larger white change areas, conglomerating the large white areas.

2.3. Quantification and Visualization

The final mask (final_mask) shows a clean binary image displaying only the confirmed, major changes.

- **Quantification:** The percentage of change is calculated as follows: (% Change) = (Total white pixels in final_mask / Total image pixels) * 100.
- **Visualization:** To visualize the results, our application creates a 6-panel report (see Figure 1) with a Change Overlay. The Change Overlay is generated by locating the contours (cv2.findContours) of the final_mask and drawing them (in red) on the original color "after" image for an intuitive, obvious visualization of where the change was detected.

3. Results: The Importance of Tuning

A key finding of this project was that those parameters of the pipeline were more important than the pipeline itself. An untuned pipeline gives misleading results.

In the case of our Pyongyang data, an initial run with a low CHANGE_THRESHOLD (30) and small KERNEL_SIZE (5x5) output a change detection result of 63.72%. Visually, this was determined to be not correct, as the change mask was fairly polluted by thousands of noisy pixels from changes associated with slight shadow and road surface differences.

Following the initial run, an iterative tuning process for the parameters was conducted utilizing the GUI sliders:

a) CHANGE_THRESHOLD from 30 to 50 : 49.52% (less noise)

b) KERNEL_SIZE from 5x5 to 7x7 : 44.55% (noise reduced)

Final parameters began with: CHANGE_THRESHOLD = 70 and KERNEL_SIZE = 9 x 9

The final, tuned system produced a result of 30.98% overall change (see Figure 1). The final_mask was visually clean, and appropriately isolated the two primary construction footprints while filtering the surrounding noise from change detection outputs - as such, 30.98% was acceptable and defensible quantification of the most substantive structural changes within the frame.

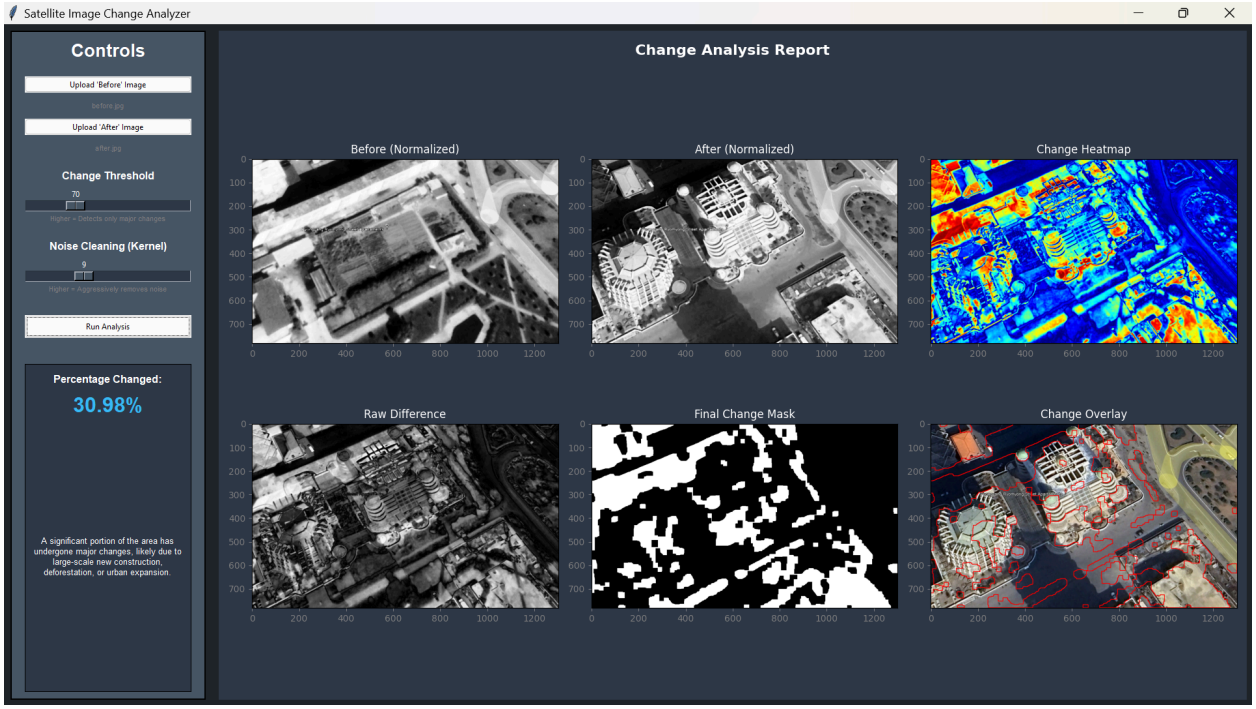

Figure 1: The final 6-panel output from the analyzer application, showing the tuned result (30.98%) for the Pyongyang case study. The "Final Change Mask" (bottom-middle) clearly isolates the new building footprints.

4. Discussion

The principal result of this paper is that our tuned pipeline (63.72% noise -> 30.98% signal) validated our hypothesis that a classical pipeline can be very helpful for high-contrast change detection if tuned properly. The human-in-the-loop, or whoever operates those THRESHOLD and KERNEL_SIZE sliders, plays an essential role in viewing the image and letting the algorithm know what "signal" and "noise" are.

4.1. Limitations

This technique is not something that will solve all problems. Its main drawbacks include:

- **Low-Contrast Change:** It would not handle subtle changes very well, such as slow-growing, perennial vegetation, slow erosion, or distinguishing between different crop types.
- **Sensitivity to Registration:** The cv2.absdiff() function is sensitive to registration. Our simple resize worked for this dataset, but an ideal solution for poor registration would involve matching features (e.g. SIFT, ORB), and warping the images to perfectly overlap them.
- **Shadows:** New, tall structures create large shadows, which could be classified as "change." We fixed this to an extent with our tuning (high threshold, large kernel), but it remains a challenge.

4.2. Classical vs. Deep Learning

While we could train a deep learning U-Net model to address this issue, it would require a large and pixel-perfect labeled dataset of "new construction." By comparison, our classical method runs completely training-free. The parameters, by design, are adapted in real-time, thereby producing a quick and highly interpretable analysis tool to assess new images paired with previous ones.

5. Conclusion

In this project, we successfully designed, developed, and tested a classical computer vision based analytical pipeline to quantify urban change. Using a high-contrast case study in Pyongyang, we demonstrated that our analysis method using Normalization -> Differencing -> Thresholding -> Morphological Refinement offers a proven and transparent workflow for assessment.

The most important finding is how important tuning parameters are. By tuning the threshold and kernel size, we were able to efficiently eliminate "noise," from small environmental changes, to detect the "signal," from major construction resulting in a reduction from an erroneous 63.72% detection of construction to a valid 30.98% detection. The final tkinter application embodies the entire pipeline presented here, which in all respects offers an effective customized first-pass analysis tool for anyone needing to perform rapid geospatial change analysis, or just looking for work on agriculture, urbanization, etc.

6. References

1. Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing (4th ed.)*. Pearson.
2. Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.
3. Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3), 90-95.
4. Harris, C.R., Millman, K.J., van der Walt, S.J. et al. (2020). Array programming with NumPy. *Nature*, 585, 357–362.
5. Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., ... & Yu, T. (2014). scikit-image: image processing in Python. *PeerJ*, 2, e453.