Laporan Projek UAS
Georeferenced Image Stitching

Disusun Oleh :

Nama : Sanvic Dicaprio
NIM : 09011282227081
Kelas : SK5C

Dosen Pengampu :

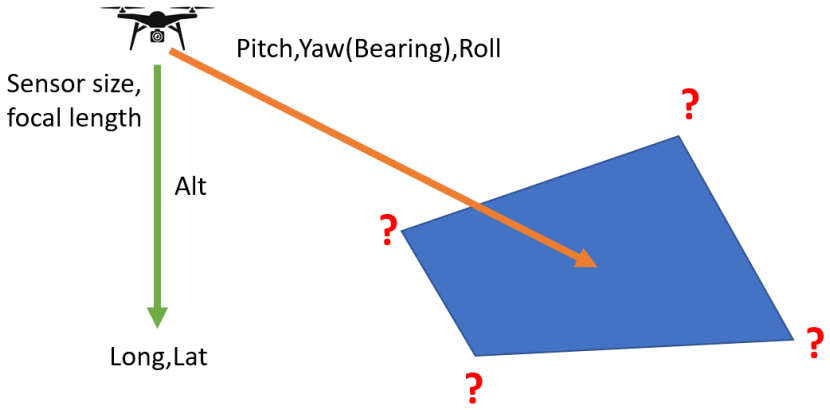
Adi Hermansyah, S.Kom., M.T.

Fakultas Ilmu Komputer
Jurusan Sistem Komputer
Universitas Sriwijaya Tahun 2023-2024

Latar Belakang Projek

Image stitching merupakan proses menggabungkan beberapa gambar menjadi satu gambar panorama. Sementara GIS (*Georeferenced Image Stitching*) adalah *image stitching* yang menggunakan informasi tambahan yaitu GPS pada data gambar tersebut. Program pada GIS memiliki kemampuan untuk dapat memperhitungkan nilai spasial seperti koordinat lintang (*latitude*), bujur (*latitude*), dan ketinggian (*altitude*) yang tersimpan dalam format EXIF (*Exchangeable Image File Format*). Biasanya proses ini melibatkan sumber gambar yang diambil dari ketinggian tertentu dengan menggunakan alat bantu seperti drone. Hasil dari *image stitching* ini yaitu sebuah panorama yang memiliki visual secara vertikal dan menyerupai lokasi aslinya.

GIS memiliki peran yang luas dalam aplikasi pemetaan gambar digital, survei lapangan, dan peranan berbasis lokasi lainnya. Pemanfaatan data GPS memungkinkan panorama yang diproses *image stitching* digunakan dalam konteks geografis yang lebih luas dan membuka potensi untuk pemahaman mendalam tentang lokasi dalam format visual. Beberapa bidang yang telah diketahui menggunakan GIS yaitu bidang pertanian, infrastruktur dan konstruksi, serta teknologi informasi satelit. Oleh karena itu, GIS membutuhkan teknik-teknik khusus untuk menangan tantangan seperti perbedaan sudut pandang, distorsi geometris, variasi pencahayaan, dan ketidaksesuaian struktur pada gambar.



Gambar 1. Visual penggunaan drone untuk mengambil gambar dengan detail GPS

Tujuan Projek

Tujuan pembuatan projek kali ini yaitu :

1. Untuk mengetahui bentuk panorama yang dibuat dengan informasi GPS dalam gambar input.
2. Untul mengetahui perbedaan antara panorama biasa dengan panorama dari GIS.

Metode Projek

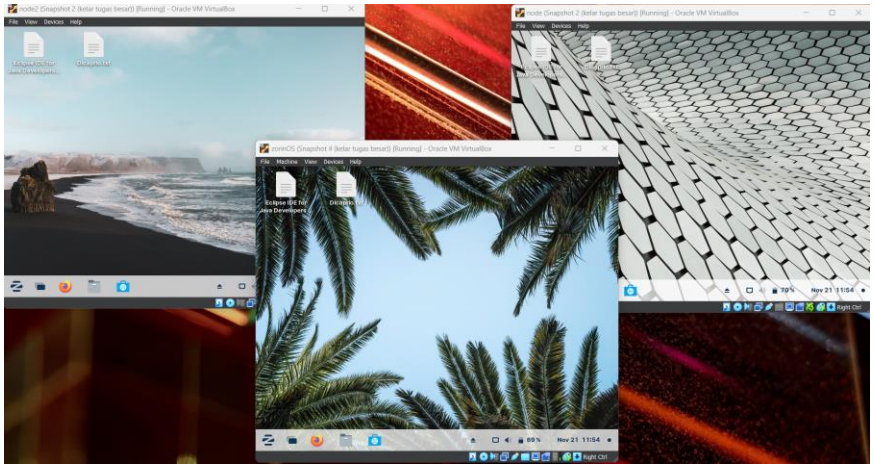
Projek ini dibuat dengan menggunakan pemrograman python yang mengekstrak informasi GPS pada EXIF dengan library piexif. Pertama, program akan membaca informasi dari nilai latitude, longtitude, dan altitude foto. Jika foto memiliki ketiga informasi GPS, program akan langsung merangkai gambar-gambar tersebut memakai

algoritma image stitching dengan bantuan library cv2 dan glob. Program python akan dijalankan secara paralel menggunakan MPI dengan 3 VM (*Virtual Machine*) dari Oracle VirtualBox berbasis GUI (*Graptical User Interface*) dengan spesifikasi VM sebagai berikut :

- Jenis sistem operasi : Zorin OS 16.3 x86_64
- Prosessor : 12th Gen Intel i5-12500H 2 Core 3.110GHz
- RAM Virtual : 1024 MB
- Jaringan : Host-Only Adapter

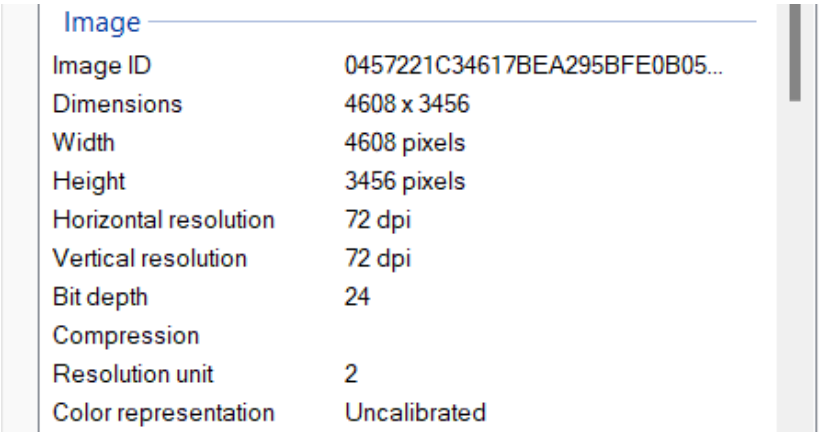
```python
from mpi4py import MPI
import cv2
import argparse
import glob
import piexif

def get_lat_lon_alt(exif):
    if 'GPS' not in exif:
        return None, None, None

    lat_tuple = exif['GPS'][piexif.GPSIFD.GPSLatitude]
    lon_tuple = exif['GPS'][piexif.GPSIFD.GPSLongitude]
    alt_tuple = exif['GPS'][piexif.GPSIFD.GPSAltitude]

    lat = convert_to_degrees(lat_tuple)
    lon = convert_to_degrees(lon_tuple)
    alt = alt_tuple[0] / alt_tuple[1]

    lat_ref = exif['GPS'][piexif.GPSIFD.GPSLatitudeRef]
    lon_ref = exif['GPS'][piexif.GPSIFD.GPSLongitudeRef]

    lat = lat * (-1 if lat_ref.upper() == 'S' else 1)
    lon = lon * (-1 if lon_ref.upper() == 'W' else 1)

    return lat, lon, alt

def convert_to_degrees(value):
    d, m, s = value
    return d[0] / d[1] + m[0] / (m[1] * 60) + s[0] / (s[1] * 3600)

def process_image(imagePath):
    image = cv2.imread(imagePath)
    exif_data = piexif.load(imagePath)
    lat, lon, alt = get_lat_lon_alt(exif_data)

    if lat is not None and lon is not None:
        return image, lat, lon, alt
    else:
        return None

def load_images_with_gps_parallel(imagePaths):
    images_with_gps = []

    comm = MPI.COMM_WORLD
    rank = comm.Get_rank()
    size = comm.Get_size()

    chunk_size = len(imagePaths) // size
    start_index = rank * chunk_size
    end_index = start_index + chunk_size if rank != size - 1 else len(imagePaths)

    local_image_paths = imagePaths[start_index:end_index]
    local_results = []

    for imagePath in local_image_paths:
        result = process_image(imagePath)
        if result is not None:
            local_results.append(result)

    all_results = comm.gather(local_results, root=0)

    if rank == 0:
        images_with_gps = [item for sublist in all_results for item in sublist]

    return images_with_gps

def main():
    ap = argparse.ArgumentParser()
    ap.add_argument("-i", "--images", type=str, required=True,
        help="path to input directory of images to stitch")
    ap.add_argument("-o", "--output", type=str, required=True,
        help="path to the output image")
    args = vars(ap.parse_args())

    comm = MPI.COMM_WORLD
    rank = comm.Get_rank()

    if rank == 0:
        print("[INFO] Memuat gambar...")

    imagePaths = glob.glob(args["images"] + "/*.jpg")
    images_with_gps = load_images_with_gps_parallel(imagePaths)

    if rank == 0:
        if not images_with_gps:
            print("[INFO] Tidak ada data GPS dalam metadata EXIF.")
        else:
            images = [img for img, _, _, _ in images_with_gps]

            print("[INFO] Merangkai gambar...")

            if cv2.__version__.startswith('3'):
                stitcher = cv2.createStitcher()
            else:
                stitcher = cv2.Stitcher_create()

            (status, stitched) = stitcher.stitch(images)

            if status == 0:
                cv2.imwrite(args["output"], stitched)

                cv2.imshow("Stitched", stitched)
                cv2.waitKey(0)

            else:
                print("[INFO] Penggabungan gambar gagal ({})".format(status))

if __name__ == "__main__":
    main()
```

Gambar 2. Porgram python yang akan dijalankan dalam projek GIS

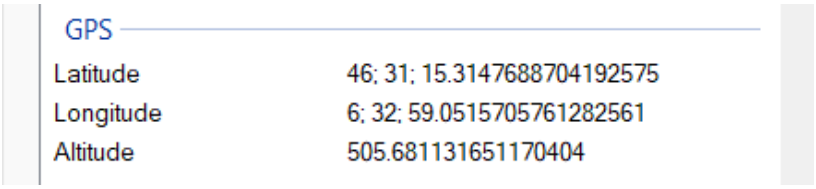


Gambar 3. VM yang akan dipakai dalam projek GIS

Untuk objek panorama, kami mengambil 50 foto dari luar yang menampilkan informasi GPS pada tahun 2016. Objek diambil menggunakan drone secara vertikal dengan ketinggian tertentu.



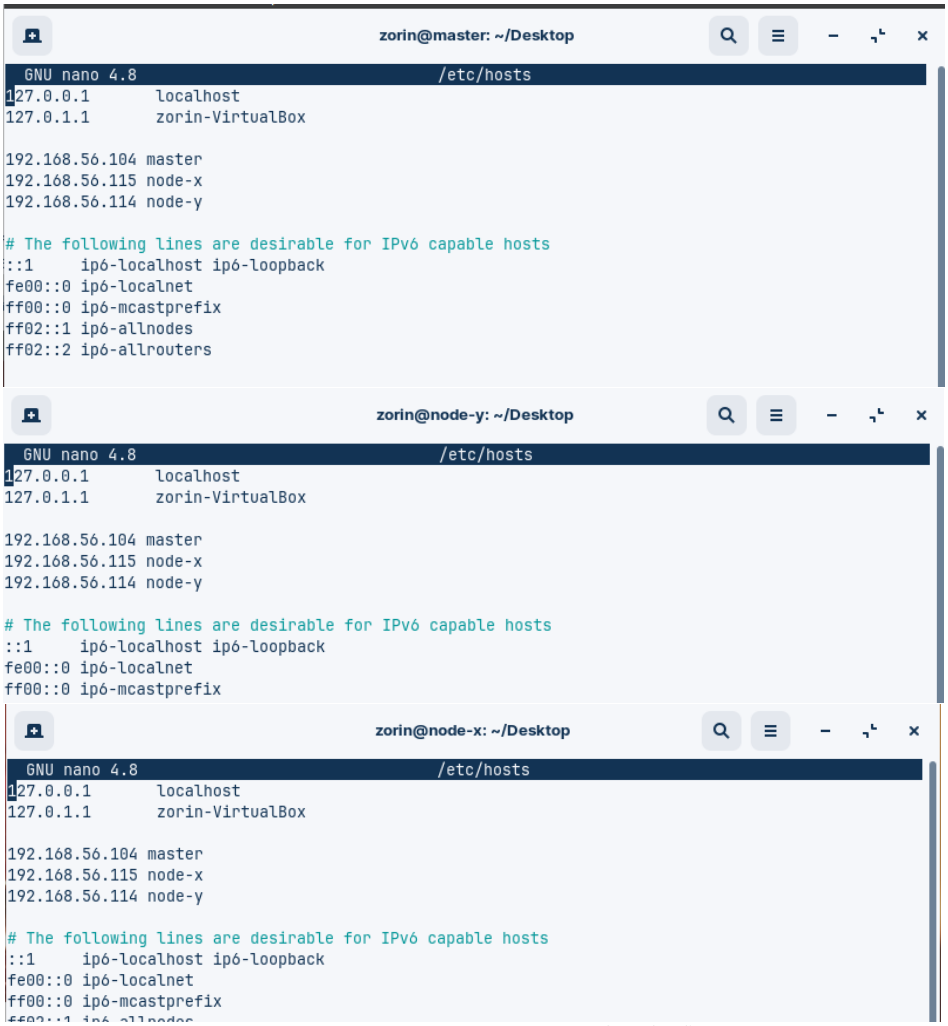
Gambar 4. Detail fisik pada foto



Gambar 5. Detail GPS pada foto

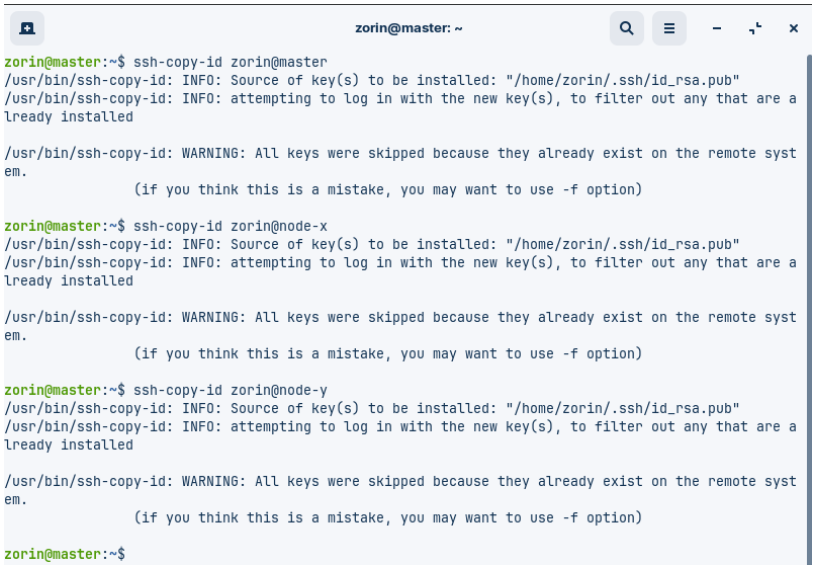
Praktikum dan Hasil Projek

1. Hidupkan ketiga VM dan masuk sebagai user MPI. User yang dipakai wajib sama antar 1 dengan VM lainnya. Hubungkan ketiga VM dengan menambah isi dari file “/etc/hosts” dengan nama hostname dan IP jaringan. Pastikan ketiga VM tersambung pada jaringan yang sama.



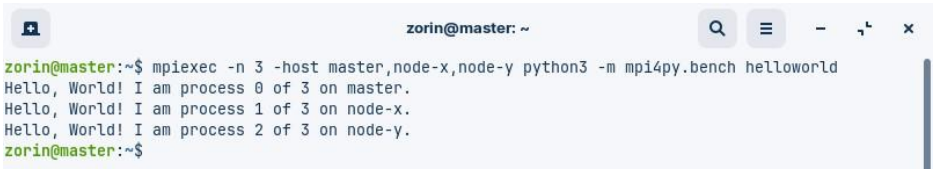
Gambar 6. Tampilan dari ketiga VM dalam file "/etc/hosts"

2. Install openssh-server di semua VM dengan "sudo apt install openssh-server". Beralih ke master (master disini yaitu zorin@master), selesaikan set up keygen dan masuk sebagai localhost. Sambungkan ketiga VM dengan "ssh-copy-id <nama_user>@<nama_hostname>".

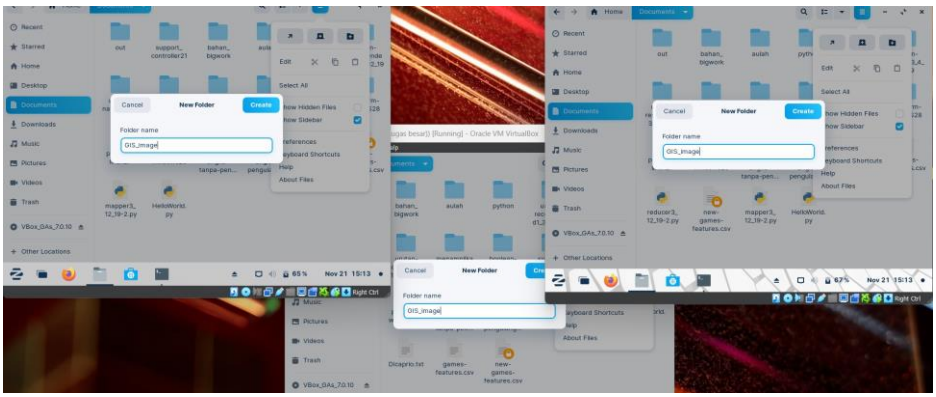


Gambar 7. Tampilan saat berhasil masuk sebagai localhost dan copy id ssh hostname di terminal

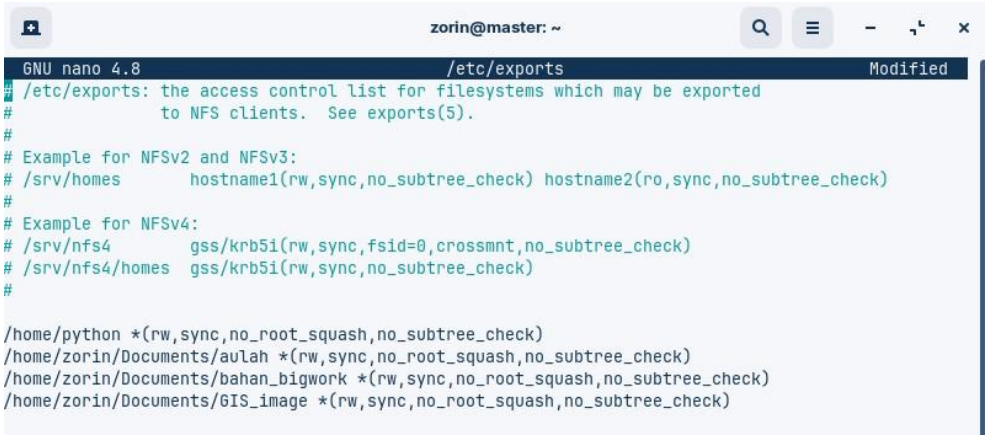
3. Cek apakah mpi telah berjalan secara paralel dengan mengetik perintah “mpiexec -n <jumlah_komputer> -host <nama_host1>,<nama_host2>,<nama_host3> python3 -m mpi4py.bench helloworld” (jumlah host tidak terlalu bergantung pada jumlah komputer. Kita dapat mengurangi jumlah komputer dibawah jumlah host namun tak dapat mengurangi jumlah host dibawah jumlah komputer).



4. Install layanan nfs agar bisa membagikan folder dengan “sudo apt install nfs-kernel-server”. Buatlah folder pada masing-masing VM dengan nama dan direktori yang sama.



5. Beralih ke terminal master, ketik “sudo nano /etc/exports”. Tambahkan kalimat dibawahnya dengan “<lokasi folder> *(rw,sync,no_root_squash,no_subtree_check)”.

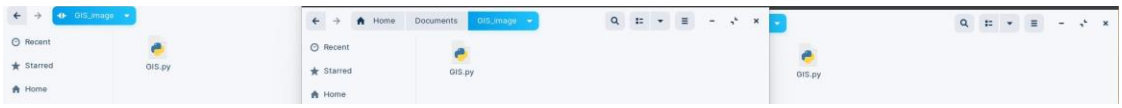


6. Simpan dan kembali. Ketik “sudo exportfs -a” untuk mengekspor ulang semua folder yang ingin dibagikan. Setelah itu, ketik “sudo systemctl restart nfs-kernel-server” untuk mereset sistem service nfs.

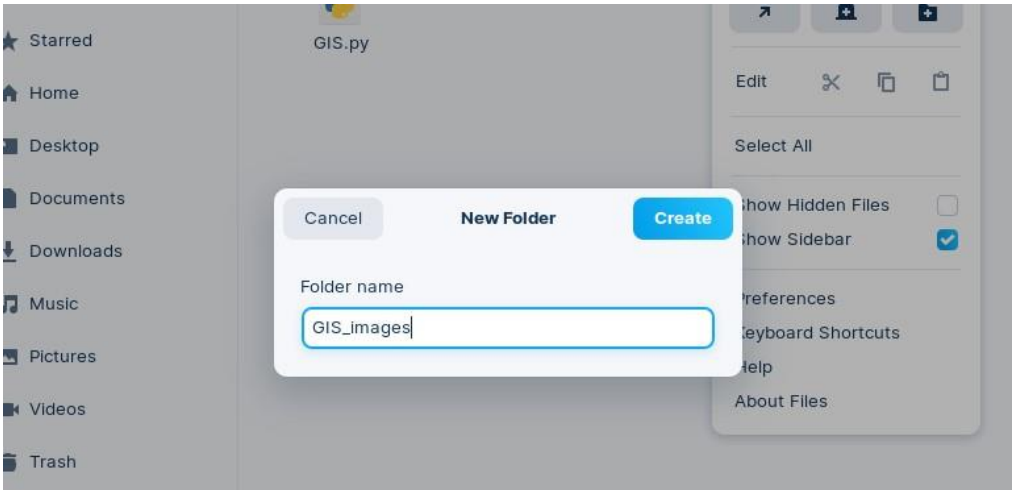
7. Kembali pada 2 VM lain. Ketik “sudo mount <Hostname_Master>:<lokasi folder di master> <lokasi folder di VM>”.

Gambar 8. Menghubungkan folder yang dibagikan master melalui layanan NFS

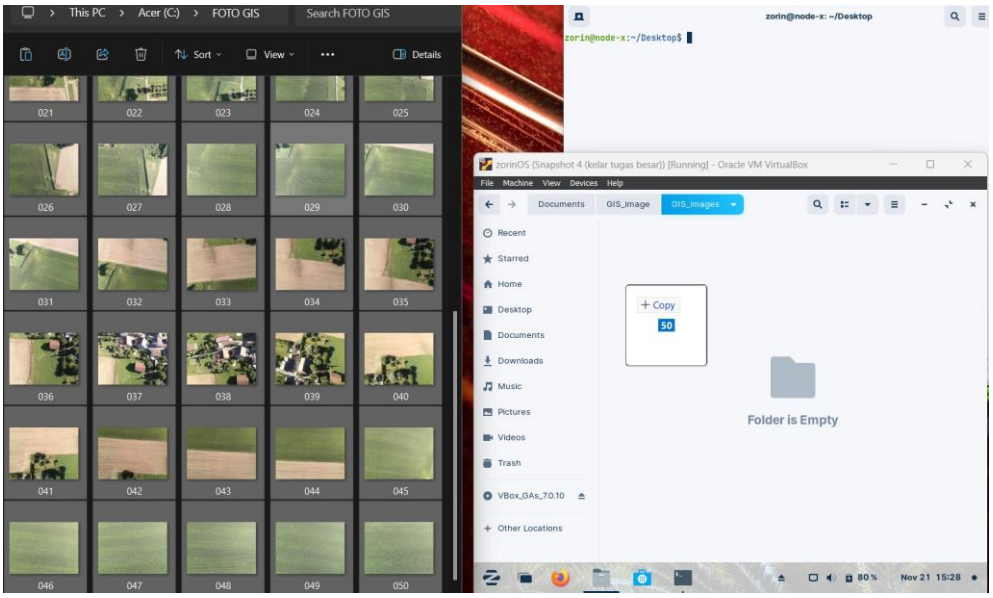
8. Cek dengan membuat file pada salah satu VM. Jika berhasil, maka akan tampil pada VM lain. (disini saya mengimpor codingan proyek GIS dengan nama GIS.py)



9. Buat folder yang akan menampung gambar projek GIS.



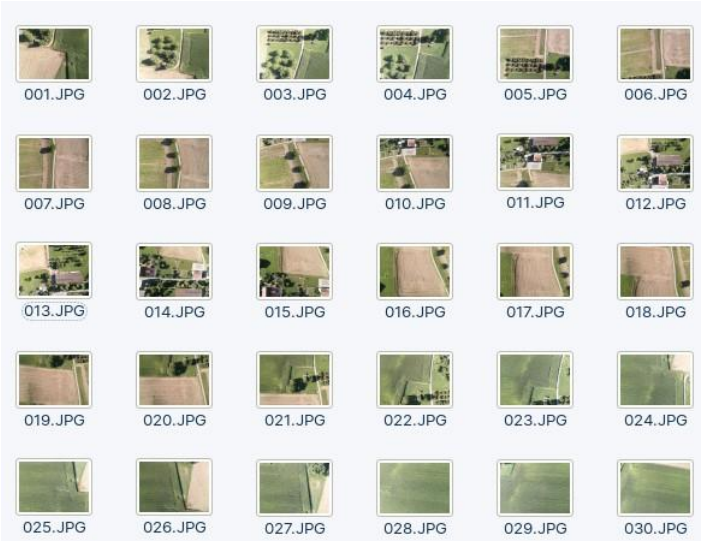
10. Ambil semua gambar dengan cara “drag and drop” semua gambar ke salah satu VM.



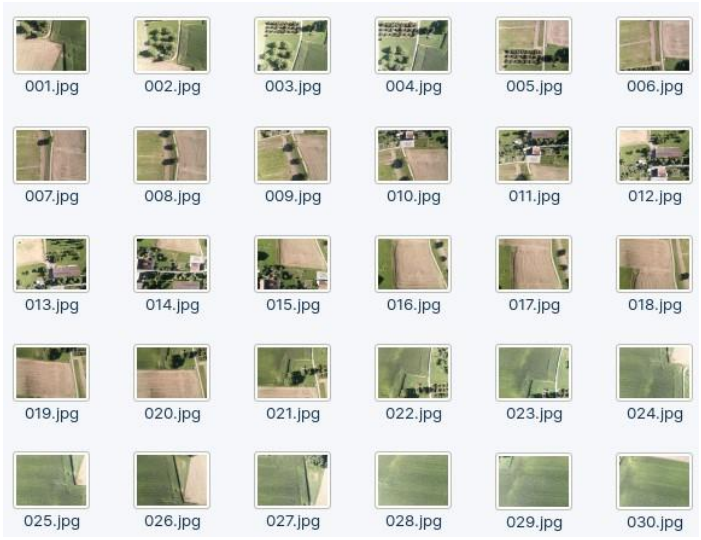
11. Samakan semua resolusi foto untuk memperkecil kemungkinan format yang error.

12. Pastikan untuk mengubahnya menjadi *.jpg. Seringkali format *.jpg saat di pindahkan dalam linux menjadi *.JPG. Hal ini mengakibatkan program tak dapat membaca EXIF pada gambar.

Gambar 9. Data EXIF tidak terbaca jika format gambar tidak diubah ke *.jpg



Gambar 10. Format gambar sebelum diubah



Gambar 11. Format gambar setelah diubah

Gambar 12. Proses mengubah format dari *.JPG ke *.jpg

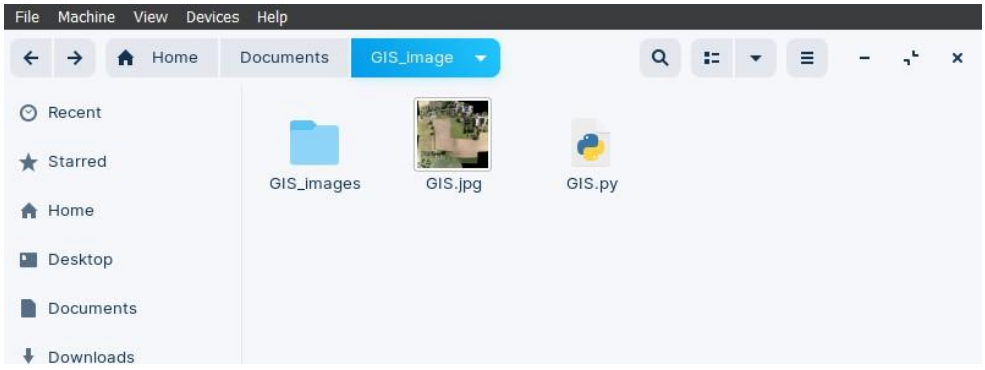
13. Untuk memastikan apakah data GPS dalam EXIF ada, kita dapat memakai tool exif dengan mengetik “exiftool <nama_gambar>”

Gambar 13. Tool untuk mengetahui data EXIF

Gambar 14. Perintah exiftool setelah dieksekusi

14. Jika semua telah dilakukan, jalankan program python. Pastikan semua VM tetap aktif saat memproses GIS melalui MPI.

15. Ketika selesai, cek pada direktori yang telah ditentukan output.



Gambar 15. Perintah MPI setelah dieksekusi

Gambar 16. Hasil gambar dari “GIS.jpg”

Perbandingan GIS dan Non-GIS

Setelah melakukan praktikum projek diatas. Kami akan membandingkan panorama dari 25 foto yang sama antara menggunakan metode GIS dan metode Non-GIS. Hal ini bertujuan untuk mengetahui perbedaan secara langsung antara panorama biasa dan panorama GIS. Program Non-GIS merupakan program python yang telah dipakai sebelumnya pada laporan 5 yaitu “image stitching.”. Program akan dijalankan pada 1 VM untuk menghilangkan dugaan terkait output yang berbeda antara pemrosesan paralel atau serial.

```python
from imutils import paths
import numpy as np
import argparse
import imutils
import cv2

ap = argparse.ArgumentParser()
ap.add_argument("-i", "--images", type=str, required=True,
        help="path to input directory of images to stitch")
ap.add_argument("-o", "--output", type=str, required=True,
        help="path to the output image")
args = vars(ap.parse_args())

print("[INFO] loading images...")
imagePaths = sorted(list(paths.list_images(args["images"])))
images = []

for imagePath in imagePaths:
        image = cv2.imread(imagePath)
        images.append(image)

print("[INFO] stitching images...")
stitcher = cv2.createStitcher() if imutils.is_cv3() else cv2.Stitcher_create()
(status, stitched) = stitcher.stitch(images)

if status == 0:
        # write the output stitched image to disk
        cv2.imwrite(args["output"], stitched)

        # display the output stitched image to our screen
        cv2.imshow("Stitched", stitched)
        cv2.waitKey(0)

else:
        print("[INFO] image stitching failed ({})".format(status))
```

Gambar 17. Isi program python dari Non-GIS

Berikut adalah 2 panorama yang menggunakan GIS dan Non-GIS :

Gambar 18. Panorama dengan metode GIS

Gambar 19. Panorama dengan metode Non-GIS

Menurut kami, cukup sulit membedakan keduanya apabila dilihat sekilas. Hal itu karena panorama baik GIS atau Non-GIS mempunyai ciri khas yang tidak menonjol Apalagi jika kedua panorama ini saling menggantikan atau menutupi. Namun setelah dilihat secara seksama dan memperbesar gambar, ada beberapa momen dimana GIS terlihat lebih unggul dibandingkan dengan Non-GIS. Begitu juga sebaliknya. Berikut beberapa perbedaan dari hasil panorama yang telah diperlihatkan :

1. **Jalan**

Gambar 20. Perbandingan 1 GIS (Kiri) dan Non-GIS (kanan)

Terlihat pada gambar tersebut, program Non-GIS mengira bahwa garis jalan pertama dan garis jalan selanjutnya tidak terhubung/terputus. Membuat program tersebut harus memaksa *stitching* agar terlihat menyatu dan mengakibatkan visual jalanan yang seolah terlihat seperti "mengunung" (naik kemudian turun). Sementara pada GIS, program telah membaca informasi GPS pada gambar tersebut. Sehingga program dapat memperkirakan berapa ketinggian pengambilan foto dan sudut pengambilan. Membuat jalanan yang "mungkin" terlihat miring dapat dibuat lurus kembali. Tapi sebagai kompensasi, kondisi garis tanah di sebelah kiri pohon GIS sama seperti jalan di panorama Non-GIS.

2. Bangunan persegi panjang dekat mobil

Gambar 21. Perbandingan 2 GIS (Kiri) dan Non-GIS (kanan)

Pada gambar ini, detail bangunan dekat pohon pada Non-GIS terlihat berada disebelah utara pohon. Sementara pada GIS, bangunan seolah berada dibawah pohon. Hal yang mungkin terjadi adalah karena pada saat pengambilan gambar, drone tak sengaja berada terlalu miring pada pohon dibandingkan rumah. Memanipulasi sudut kemiringan bujur dan lintang dalam pengambilan dan pemrosesan panorama GIS. Namun Non-GIS

memprosesnya seperti 2 bangunan yang berbeda. Sehingga membuat 2 perspektif pada gambar terpecah saat pembuatan panorama.

3. **Detail kecil pada tempat tertentu**

Gambar 22. Perbandingan 3 GIS (Kiri) dan Non-GIS (kanan)

Gambar 23. Perbandingan 4 GIS (Kiri) dan Non-GIS (kanan)

Detail seperti pagar tanaman (bawah) dalam panorama Non-GIS tetap ditampilkan, sementara pada GIS dihilangkan. Lalu karena point 1 yang membahas detail jalanan pada GIS dioptimalkan berdasarkan perhitungan, ia juga mengorbankan tingkat presisi pada garis batas jalan. Membuat jalan tersebut terlihat patah / miring.

Kesimpulan

GIS (*Georeferenced Image Stitching*) adalah proses penggabungan gambar membentuk panorama yang memakai nilai GPS dalam menentukkan hasil akhir. Oleh karena itu, GIS lebih sering dipakai pada bidang yang sering berhubungan dengan data geografi. GIS juga membutuhkan tingkat presisi yang bagus pada alat bantu seperti drone agar tetap stabil dalam pengambilan objek. Perbedaan paling mendasar GIS dan Non-GIS adalah Non-GIS lebih memprioritaskan visual dengan efek panorama yang lebih enak diliat dalam 1 arah. Sementara GIS lebih memprioritaskan hasil akhir yang sesuai dengan perhitungan masing-masing latitude, longtitude, dan altitude pada semua gambar. Sehingga beberapa detail

yang kurang banyak muncul dalam foto akan dihilangkan agar panorama GIS terlihat lebih mirip dengan lokasi aslinya. Makan oreo bersama kekasih, saya dicaprio dan terimakasih. Sfx* teng teng teng teng teng teng teng…