HOMEWORK 18

Image Matching

Student Name: Rohit Das **Student ID**: 61047086s

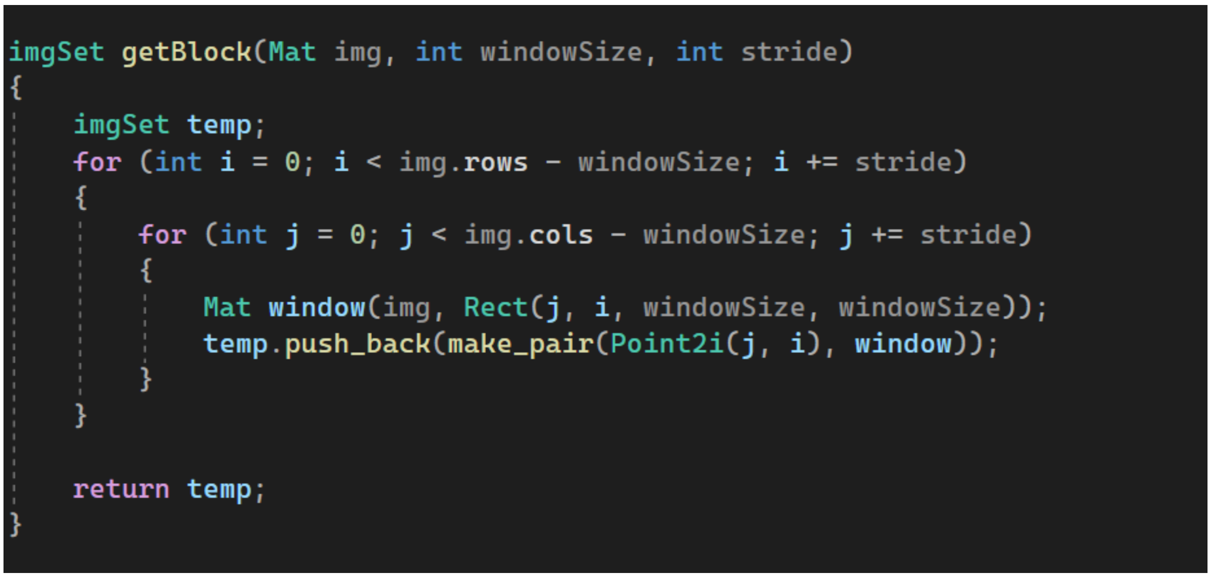
Objective: Detecting Motion Vectors in two images.

Algorithm Used: Traditional Algorithm

- Create the Windowing Function.
- Compare the two image pixel values on the same window.
- Find the Direction.
- Apply Quiver on the image.

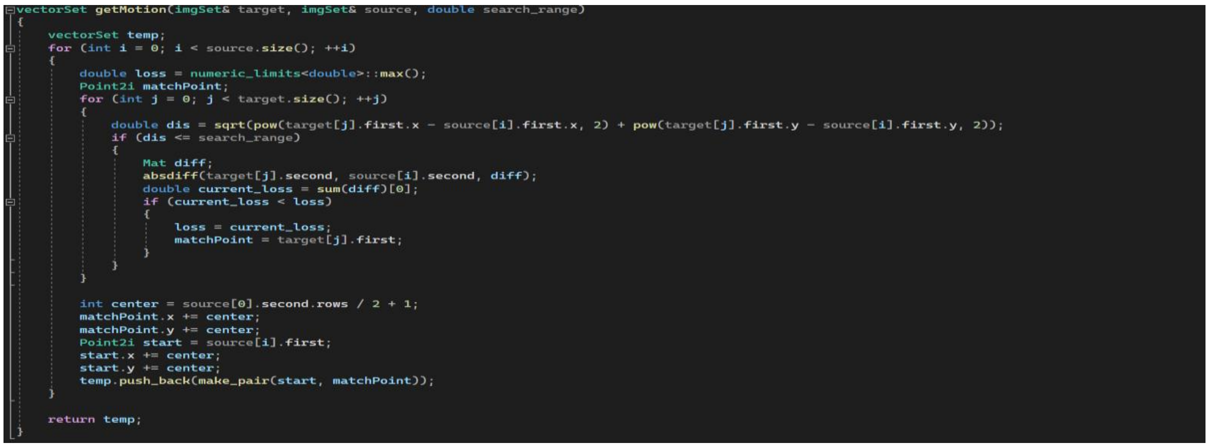
Screenshot of the Algorithm:

Create the Window:



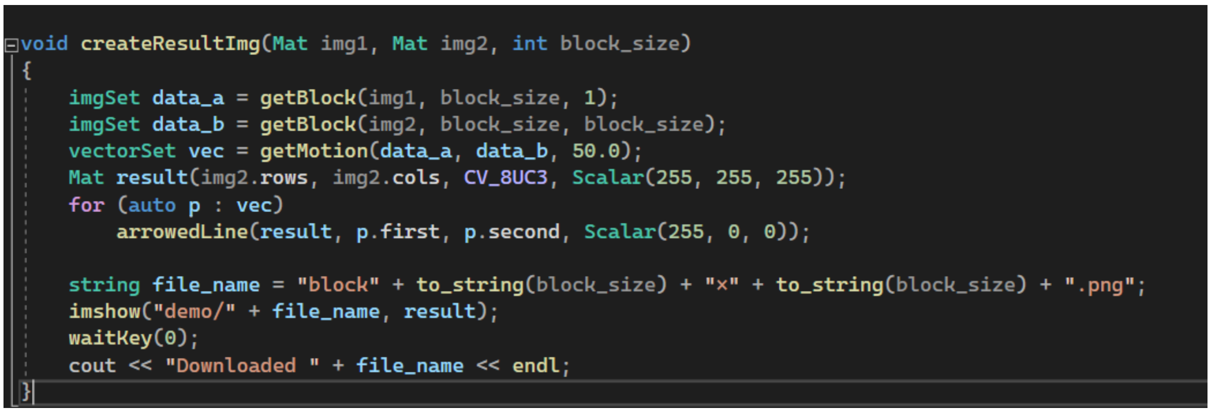
```cpp
imgSet getBlock(Mat img, int windowSize, int stride)
{
    imgSet temp;
    for (int i = 0; i < img.rows - windowSize; i += stride)
    {
        for (int j = 0; j < img.cols - windowSize; j += stride)
        {
            Mat window(img, Rect(j, i, windowSize, windowSize));
            temp.push_back(make_pair(Point2i(j, i), window));
        }
    }

    return temp;
}
```

Find the direction of motion



```cpp
vectorSet getMotion(imgSet& target, imgSet& source, double search_range)
{
    vectorSet temp;
    for (int i = 0; i < source.size(); ++i)
    {
        double loss = numeric_limits<double>::max();
        Point2i matchPoint;
        for (int j = 0; j < target.size(); ++j)
        {
            double dis = sqrt(pow(target[j].first.x - source[i].first.x, 2) + pow(target[j].first.y - source[i].first.y, 2));
            if (dis <= search_range)
            {
                Mat diff;
                absdiff(target[j].second, source[i].second, diff);
                double current_loss = sum(diff)[0];
                if (current_loss < loss)
                {
                    loss = current_loss;
                    matchPoint = target[j].first;
                }
            }
        }

        int center = source[0].second.rows / 2 + 1;
        matchPoint.x += center;
        matchPoint.y += center;
        Point2i start = source[i].first;
        start.x += center;
        start.y += center;
        temp.push_back(make_pair(start, matchPoint));
    }

    return temp;
}
```

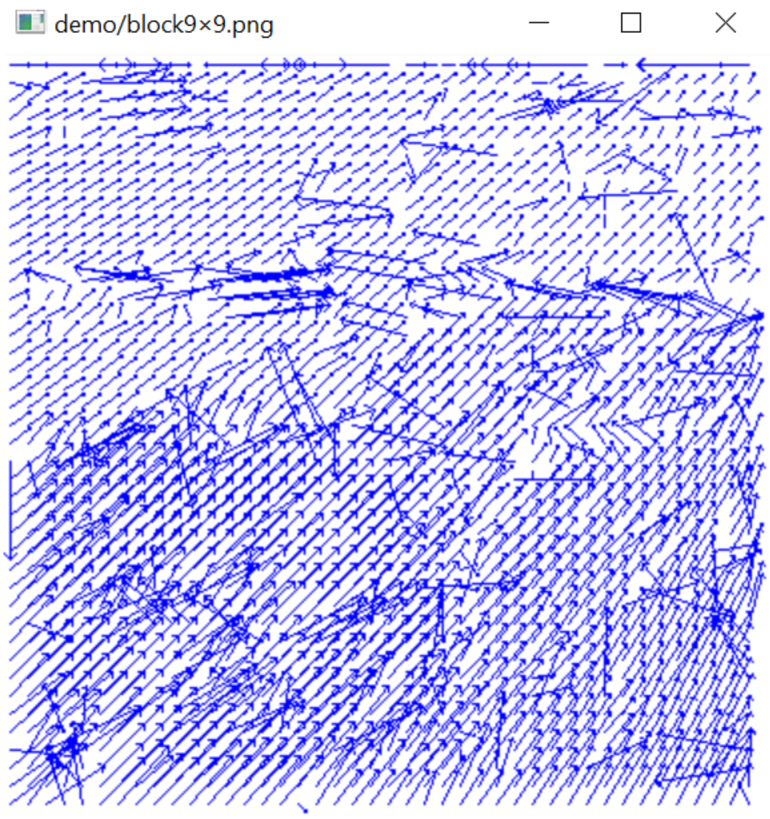
Create the motion Diagram



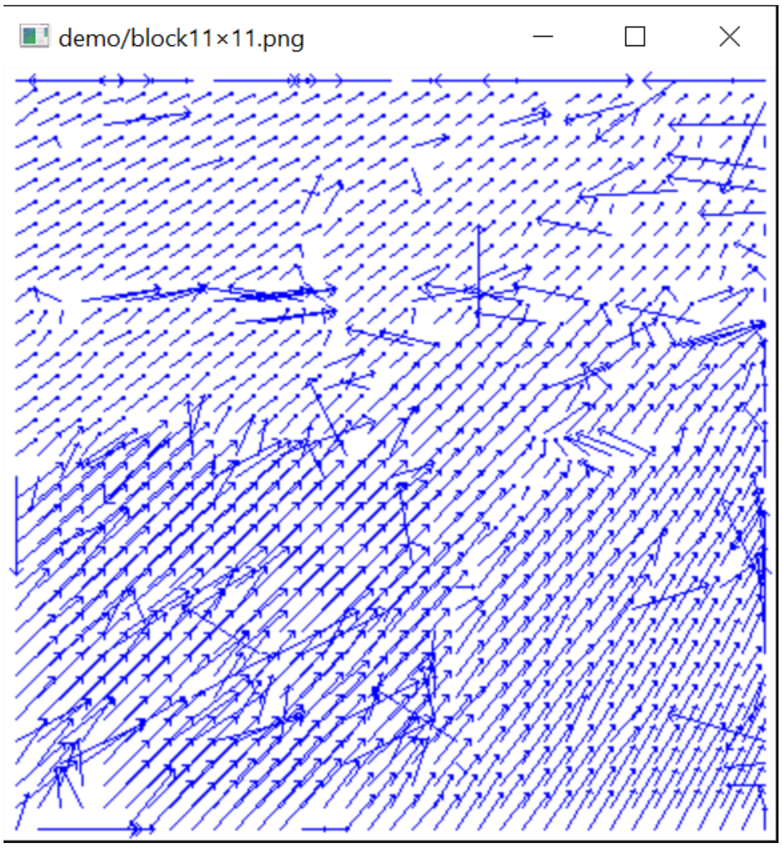
```
void createResultImg(Mat img1, Mat img2, int block_size)
{
    imgSet data_a = getBlock(img1, block_size, 1);
    imgSet data_b = getBlock(img2, block_size, block_size);
    vectorSet vec = getMotion(data_a, data_b, 50.0);
    Mat result(img2.rows, img2.cols, CV_8UC3, Scalar(255, 255, 255));
    for (auto p : vec)
        arrowedLine(result, p.first, p.second, Scalar(255, 0, 0));

    string file_name = "block" + to_string(block_size) + "x" + to_string(block_size) + ".png";
    imshow("demo/" + file_name, result);
    waitKey(0);
    cout << "Downloaded " + file_name << endl;
}
```

Example Images

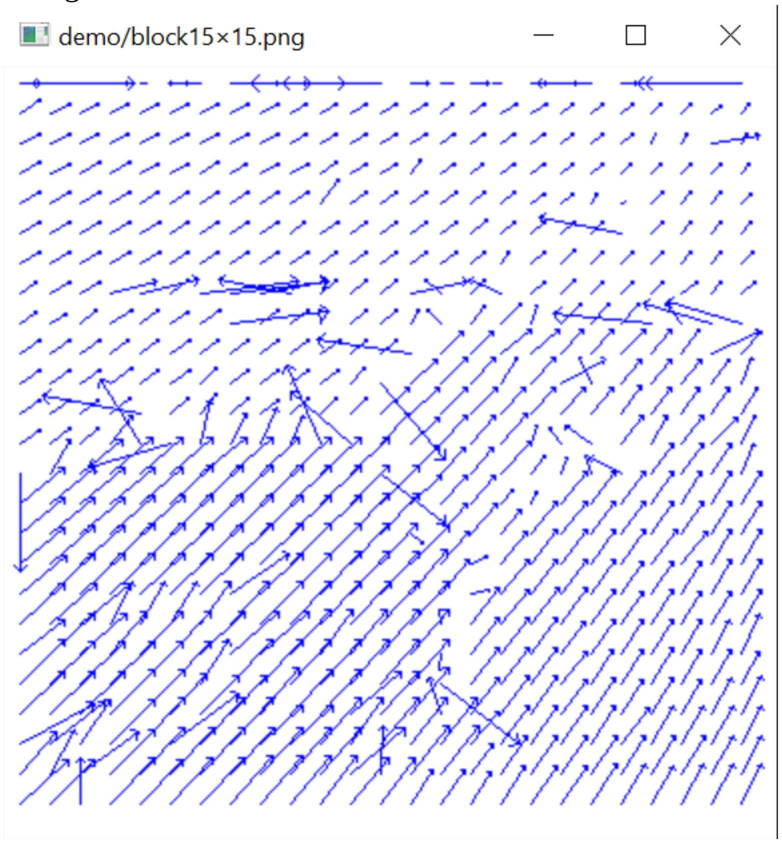
1. Using 9*9 Block



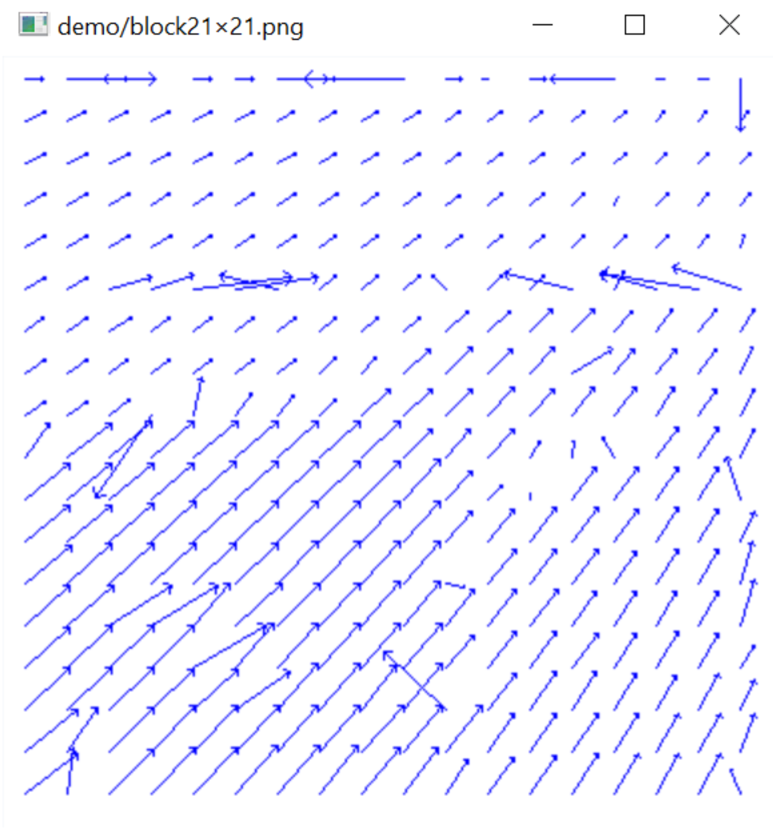
2. Using 11*11 Block



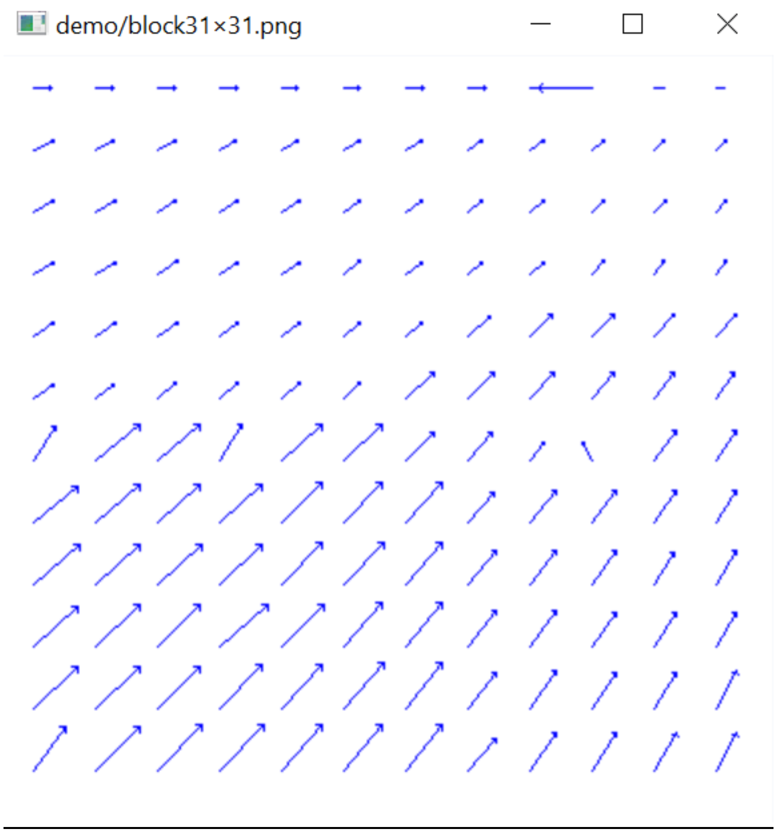
3. Using 15*15 Block



4. Using 21*21 Block



5. Using 31*31 Block


demo/block31×31.png