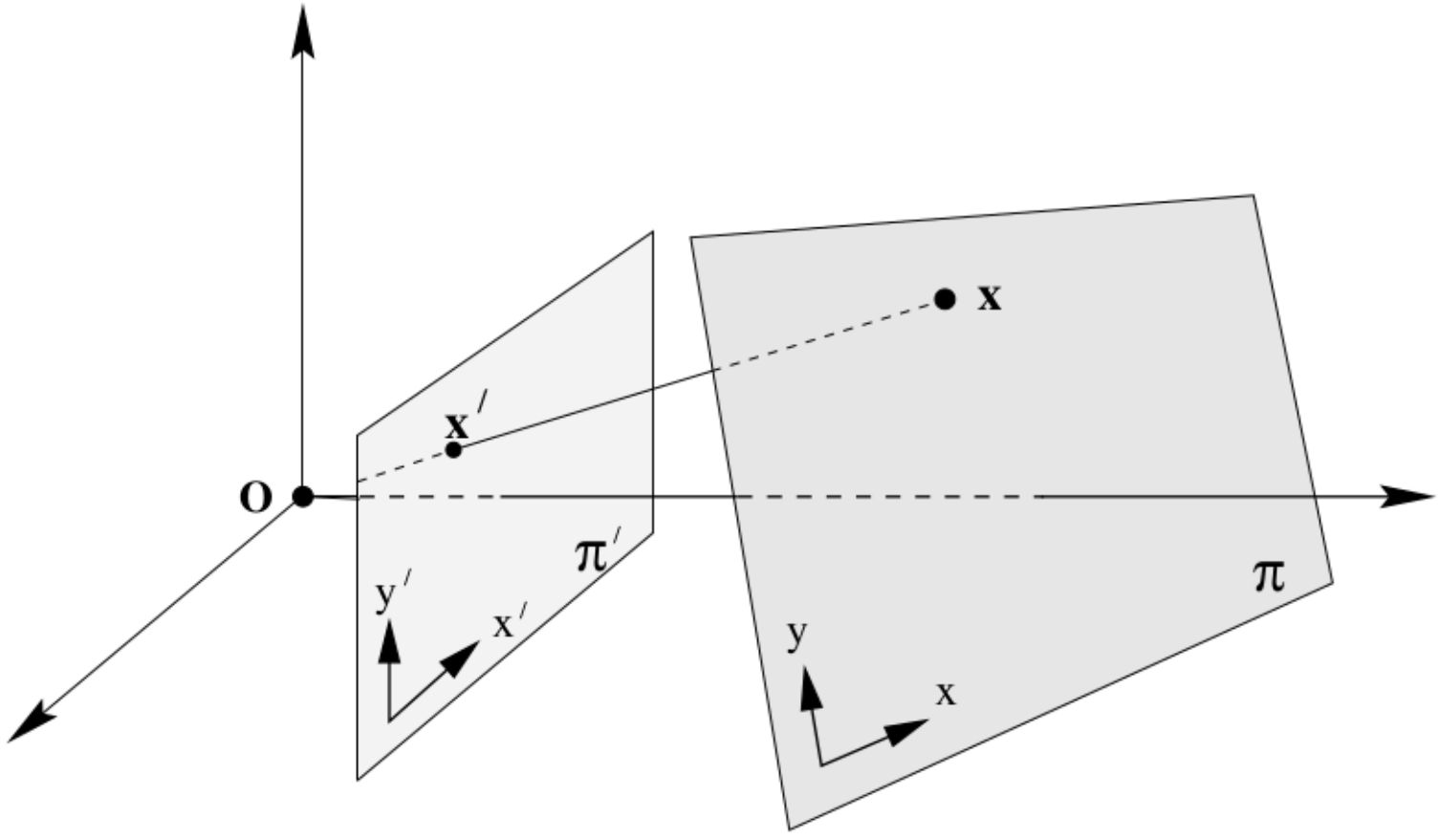
ECE 417/598: Direct Linear Transform

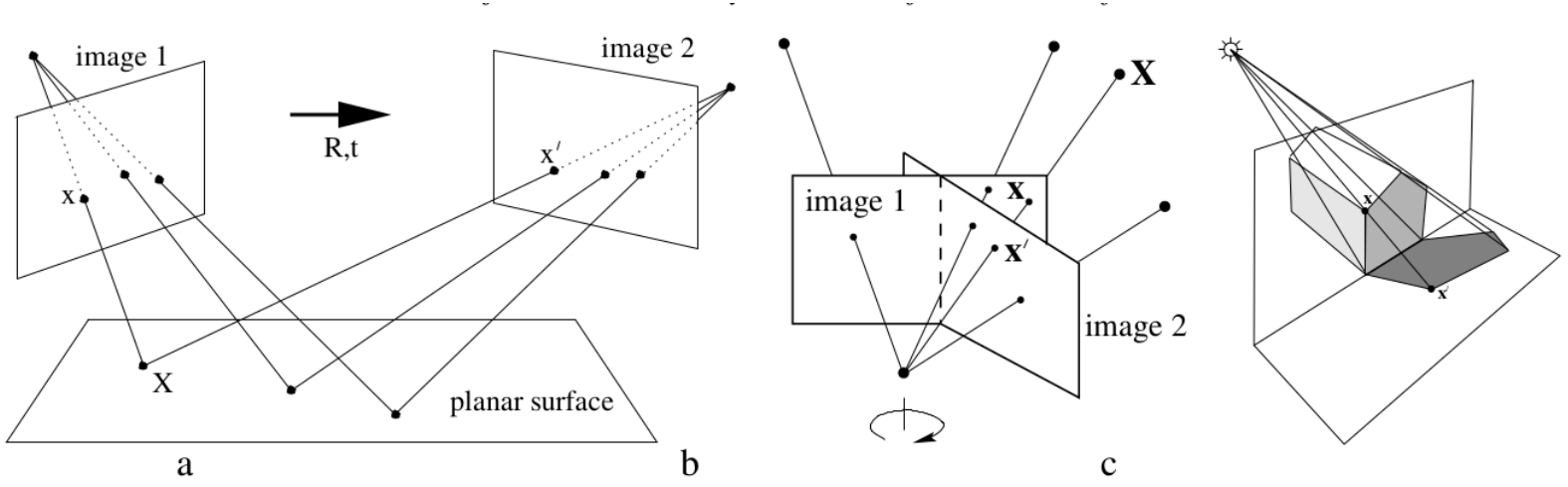
Vikas Dhiman

March 23, 2022

Homography

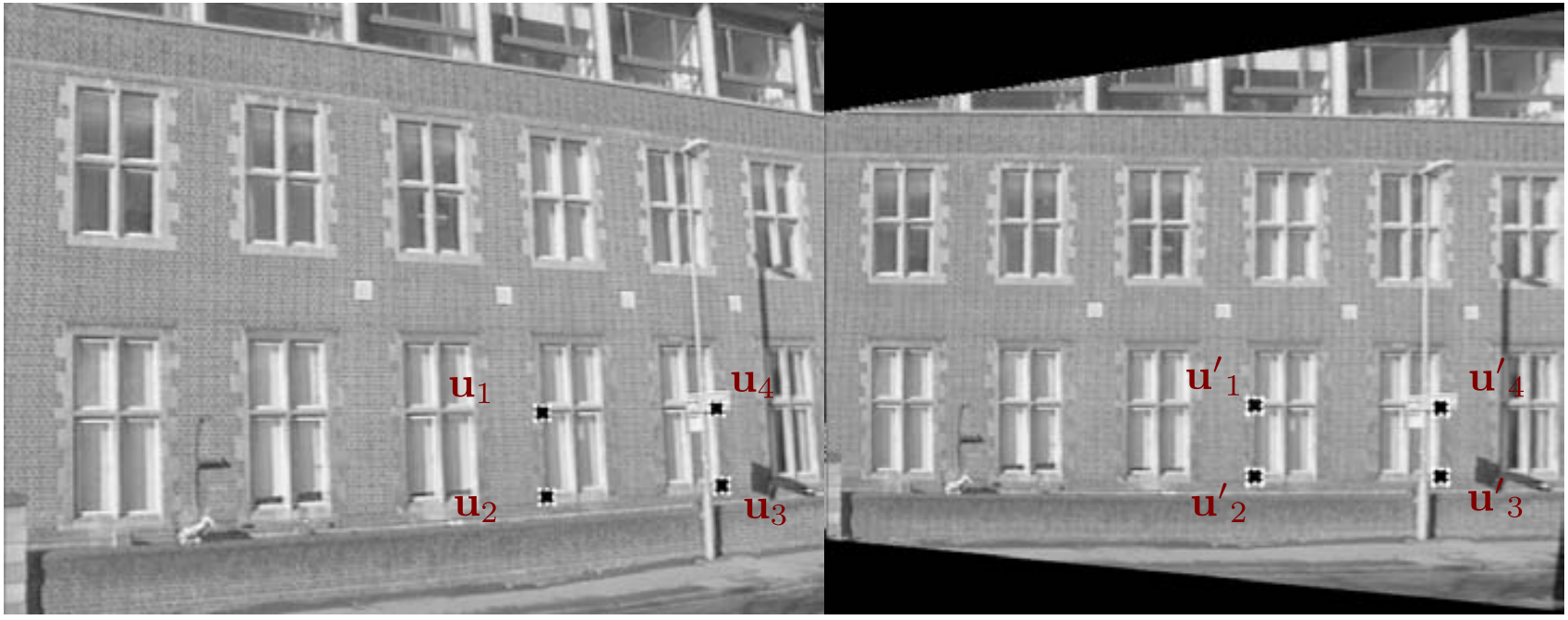


Examples of Homography




Top view
Graphic
Look! Safe to move?
Settings

Computing Homography



$$\underline{\mathbf{u}}_1 = [100, 98, 1]^\top \qquad \underline{\mathbf{u}}_2 = [102, 95, 1]^\top$$
$$\underline{\mathbf{u}}_3 = [107, 90, 1]^\top \qquad \underline{\mathbf{u}}_4 = [110, 85, 1]^\top$$
$$\underline{\mathbf{u}}'_1 = [100, 98, 1]^\top \qquad \underline{\mathbf{u}}'_2 = [102, 95, 1]^\top$$
$$\underline{\mathbf{u}}'_3 = [107, 98, 1]^\top \qquad \underline{\mathbf{u}}'_4 = [110, 95, 1]^\top$$

Find H such that $\underline{\mathbf{u}}' = H\underline{\mathbf{u}}$ for any point on one image to another image, where $\underline{\mathbf{u}}', \underline{\mathbf{u}} \in \mathbb{P}^2$

2D homography

Given a set of points $\underline{\mathbf{u}}_i \in \mathbb{P}^2$ and a corresponding set of points $\underline{\mathbf{u}}'_i \in \mathbb{P}^2$, compute the projective transformation that takes each $\underline{\mathbf{u}}_i$ to $\underline{\mathbf{u}}'_i$. In a practical situation, the points $\underline{\mathbf{u}}_i$ and $\underline{\mathbf{u}}'_i$ are points in two images (or the same image), each image being considered as a projective plane $\mathbb{P}^2$.

Solving for Homography

Solving for Homography

Solving for Homography

```
Eigen::Matrix3d
findHomography(const std::vector<Eigen::Vector3d>& us,
               const std::vector<Eigen::Vector3d>& ups)
{
    Eigen::MatrixXd A(8, 9); A.setZero();
    for (int i = 0; i < us.size(); ++i) {
        // [[0^T      -w'_i u_i^T   y_i' u_i^T]]
        //  [w_i'u_i^T        0^T   -x_i u_i^T]]
        A.block(2*i, 3, 1, 3) = -ups[i](2)*us[i].transpose();
        A.block(2*i, 6, 1, 3) = ups[i](1)*us[i].transpose();
        A.block(2*i+1, 0, 1, 3) = ups[i](2)*us[i].transpose();
        A.block(2*i+1, 6, 1, 3) = -ups[i](0)*us[i].transpose();
    }

    auto svd = A.jacobiSvd(Eigen::ComputeFullV);
    Eigen::Matrix3d H;
    Eigen::VectorXd nullspace = svd.matrixV().col(8);
    H.row(0) = nullspace.block(0, 0, 3, 1).transpose();
    H.row(1) = nullspace.block(3, 0, 3, 1).transpose();
    H.row(2) = nullspace.block(6, 0, 3, 1).transpose();

    return H;
}
```

Apply Homography

```
Eigen::MatrixXd
applyHomography(const Eigen::Matrix3d& H,
                const Eigen::MatrixXd& img) {
    Eigen::MatrixXd new_img(img.rows(), img.cols());
    Eigen::Vector3d u;
    Eigen::Vector3d up;
    for (int new_row = 0; new_row < new_img.rows(); ++new_row) {
        for (int new_col = 0; new_col < new_img.cols(); ++new_col) {
            u << new_col + 0.5, new_row + 0.5, 1;
            /**** Apply homography for each pixel ***/
            up = H * u;
            up /= up(2);
            /**** Apply homography for each pixel ***/
            int row = round(up(1));
            int col = round(up(0));
            if (0 <= row && row < img.rows()
                && 0 <= col && col < img.cols()) {
                new_img(new_row, new_col) = img(row, col);
            }
        }
    }
    return new_img;
}
```

3D to 2D camera projection matrix estimation

Given a set of points $\mathbf{X}_i$ in 3D space, and a set of corresponding points $\mathbf{x}_i$ in an image, find the 3D to 2D projective $\mathbf{P}$ mapping that maps $\mathbf{X}_i$ to $\mathbf{x}_i = \mathbf{P}\mathbf{X}_i$.