**Abstract**:

Our feed-forward network has 10 layers, takes two stacked grayscale images as input, and produces an 8 degree of freedom homography which can be used to map the pixels from the first image to the second.

**What is homography?**

transformation relating two images undergoing a rotation about the camera center is a homography

**How to make a traditional homography robust?**

1. Robustness is introduced into the corner detection stage by returning a large and over-complete set of points,
2. robustness into the homography estimation step shows up as heavy use of RANSAC or robustification of the squared loss function

**Architecture**

* VGG-style [17] network for the homography estimation task
* **Input** : (IA; IB;HAB) training triplets; **Labels** : 4pt homography i.e. the displacement in pixel coordinates
* **Our networks use 3x3 convolutional blocks with Batch-Norm [10] and ReLUs, and are architecturally similar to Oxfords VGG Net**
* Input : the two input images, which are related by a homography, are stacked channel-wise and fed into the network
* Networks:
  + The **regression network** directly produces 8 real-valued numbers and uses the Euclidean (L2) loss as the final layer during training.
  + **classification network** uses a quantization scheme. It uses softmax at the last layer, and we use the cross-entropy loss. We chose to use 21 quantization bins for each of the 8 output dimensions, which results in a final layer with 168 output neurons.
    - While quantization means that there is some inherent quantization error

**Training procedure**

**Evaluation Metric:**

To measure this metric, one first computes the L2 distance between the ground truth corner position and the estimated corner position. The error is averaged over the four corners of the image, and the mean is computed over the entire test set

**Data Creation & Augmentation:**

* **Creation:**

1. we generate a nearly unlimited number of labeled training examples by applying random projective transformations to a large dataset.
2. To generate a single training example, we first randomly crop a square patch from the larger image I at position p (we avoid the borders to prevent bordering artifacts later in the data generation pipeline). This random crop is Ip. [256\*256]
3. We perturb the corners by some value [-rho,rho]
4. We get (Hab)^-1. We transform the img1 to get warped image and also crop it at same locations as image 1

* Augmentation:
  + to make our method more robust to motion blur, we can apply such blurs to the image in our training set. If we want the method to be robust to occlusions, we can insert random occluding shapes into our training images.

**Loss**

* Regression: eucledian(L2)
* Classification: Cross entropy

**Key Ideas**

* Homography estimation problem as classification, which produces a distribution over homographies and can be used to determine the confidence of an estimated homography**.**
* **Why use 4pt param?**
  + However, if we unroll the 8 (or 9) parameters of the homography into a single vector, we will quickly realize that we are mixing both rotational and translational terms. For example, the submatrix [H11 H12; H21 H22], represents the rotational terms in the homography, while the vector is the [H13 H23] translational offset
  + Once the displacement of the four corners is known, one can easily convert H4point to Hmatrix. This can be accomplished in a number of ways, for example one can use the normalized Direct Linear Transform (DLT) algorithm [9], or the function getPerspectiveTransform()in OpenCV