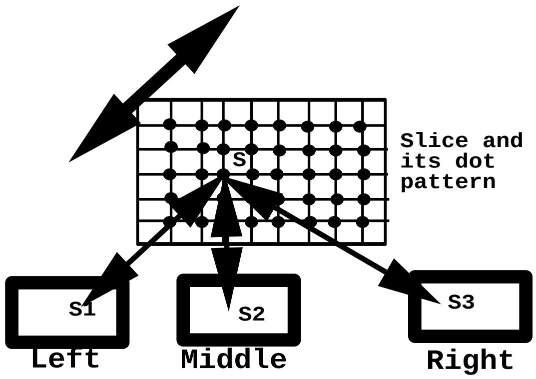


Active space creation

Figure 1: Imprint-set (S1,S2,S3) for point S. S1, S2, and S3 are 2D points on the respective camera-images.

Procedure DataCollection

1. Mark a 12x12 grid pattern on a whiteboard. Physical grid-cells are 5cm by 5cm
2. Set up three cameras so that the grid pattern is visible in all the three cameras
3. Capture the three camera-images and store them
4. Repeat step 3 seven more times, after moving the white-board with the grid pattern seven more times

end DataCollection

Figure 2: Preprocessing: Data Collection Algorithm for scanning a 55cm by 55cm by 70cm active-space.

```
Procedure CreateActiveSpaceIndexing

 1. For a set of left, center, and right camera
    images collected in step 3 of the DataCollection
    Algorithm

    do
     2. Identify all the 12x12 grid−intersection
        points
     3. Store the pixel−location of the grid−points
        along with their 3D−points on the white−board
    end do

 4. Repeat step 1−3, for all the eight sets of three
    camera−images.
end CreateActiveSpaceIndexing
```

Figure 3: (Preprocessing) Creation of the Active Space Indexing data structure.

```
Procedure FindingA2DIndex(S)
 // returns the 2D Index (p,q) given a point S on
 //  the slice

 // Let x = Sx, y = Sy
 If (x,y) is within the Slice

    1. Find two consecutive vertical grid lines p and
       p+1 so that x is between p and p+1
    2. Find two consecutive horizontal grid−lines q
       and q+1 so that y is between q and q+1
    3. return(p,q)
  endif
  // otherwise S is not within the Slice
  return (not inside the Slice)
end FindingA2DIndex
```

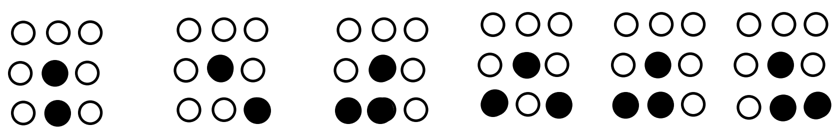
Figure 4: Returns the 2DIndex or cell which contains the given point S

```
Procedure Find_S(S1, S2, S3)

// Given the imprint Set (S1,S2, S3)
// find the 3D location S

1. Use S1 for the left image, S2 for
   the center camera-image, and S3 for
   the right-camera image

   do

2.   Find indices I1, I2, and I3 for all
     the eight slices.  I1, I2, and I3
     are the grid-cells containing S1, S2, and S3.
     Note that I1, I2, and I3 refer to the projected
     cell locations for the white board locations.
     Therefore, there are eight such possibilities.
     For some white board placements, I1=I2=I3
     and the area would be zero.  Basically we
     are looking for two consecutive whiteboard
     slices between which we expect the point S
     to lie. If there are several white-board locations
     with zero area, then we select the first with
     zero area. If such a situation does not exist
     then conclude that (S1,S2,S3) do not correspond,
     otherwise perform steps 3-4 below.

3.   Using the two consecutive slices in the active
     space, find the 3D location of point S1 on these
     two slices.  This will define a line L1.  Similarly
     obtain lines L2 and L3 using S2 and S3,
     respectively.

4.   Find the minimum distances between the three pairs
     of lines (L1, L2), (L2, L3), (L1, L3).  Take the
     average of the three distances.  If this average
     is greater than the "closeness" criteria, conclude
     that (S1,S2,S3) do not correspod, else calculate the
     nearest points on these lines.  That would define
     the location of point S in 3D. Return point S.

end Find_S
```

Figure 5: Algorithm: Find S provides estimation of point S. If S1, S2, S3 are corresponding closely then precise estimates are obtained.

Procedure SpatialMarking

1. For every pixel in all the camera images
   do
2. Identify Significant points by considering the (r,g,b) of the 8 neigboring pixels, and using thresholding. Some cases when the center pixel is identified as significant point are given below:



Note: Mirror cases will also classify the pixel as significant

3. If a pixel is identified as significant then

   For all the 8 slices
   do

4. Find the 2D Index, and add this pixel as candidate for being a significant pair for that 2D−cell indicated by the 2Dindex on a slice. Note that the 2D Index and the slice, in fact define a grid−voxel. And we have a 12x12x8 3D−grid of voxels in our implementation.
   end for
   end if
   end for

Figure 6: Identifying significant points and Marking them in the 3D grid Voxels.

```
Procedure SpatialFiltering

1. For all the 12x12x8 3D grid voxels
   do
// Let p1, p2, p3 be pixels identified as significant
// (by the SpatialMarking process) from camera 1, 2, 3

2.  for i= 1 to p1

3.     S1 = ith pixel in this grid-voxel from camera 1
4.     for j = 1 to p2

5.        S2 = jth pixel in this grid-voxel from camera 2
6.        for k = 1 to p3

7.        S3 = kth pixel in this grid-voxel from camera 3
8.        thisPoint3D = Find_S(S1,S2,S3) // See Figure 3
9.        display(thisPoint3D)

          endfor k
       endfor j
    endfor i
   endfor
```

Figure 7: Generating Corresponding-pairs and 3D points

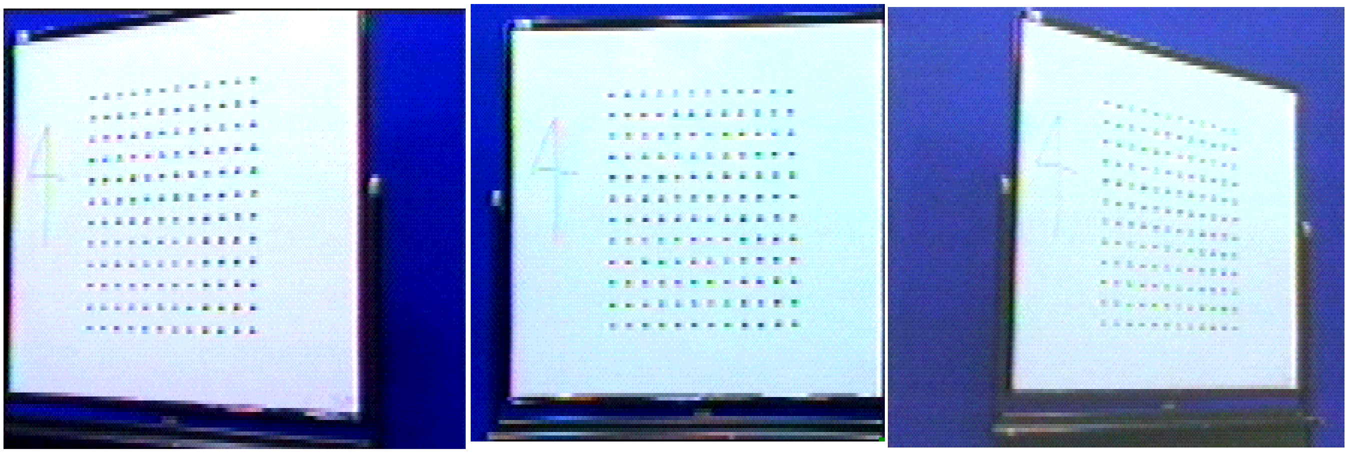


Figure 8: Whiteboard used for Data collection and three views

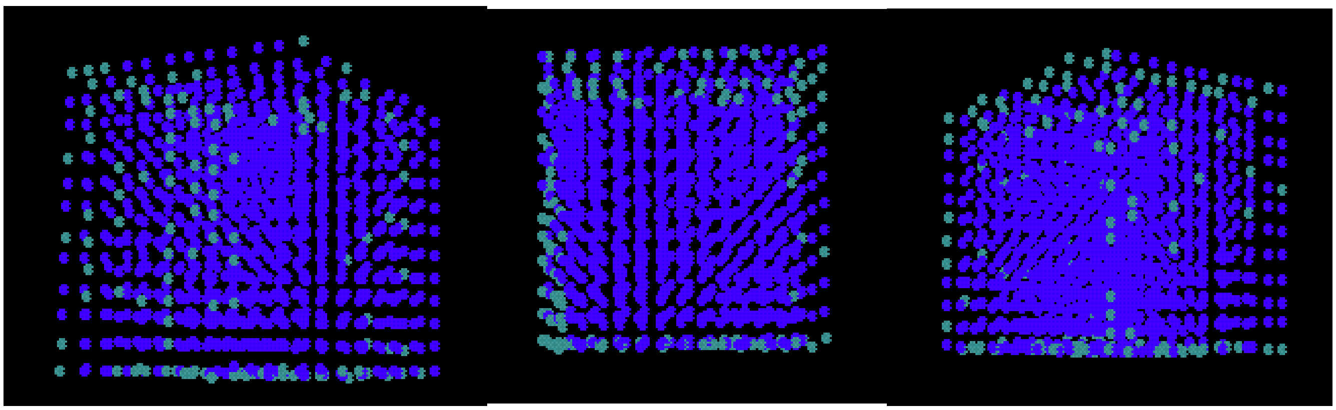


Figure 9: The projection points on all the three cameras for all whiteboard slice positions

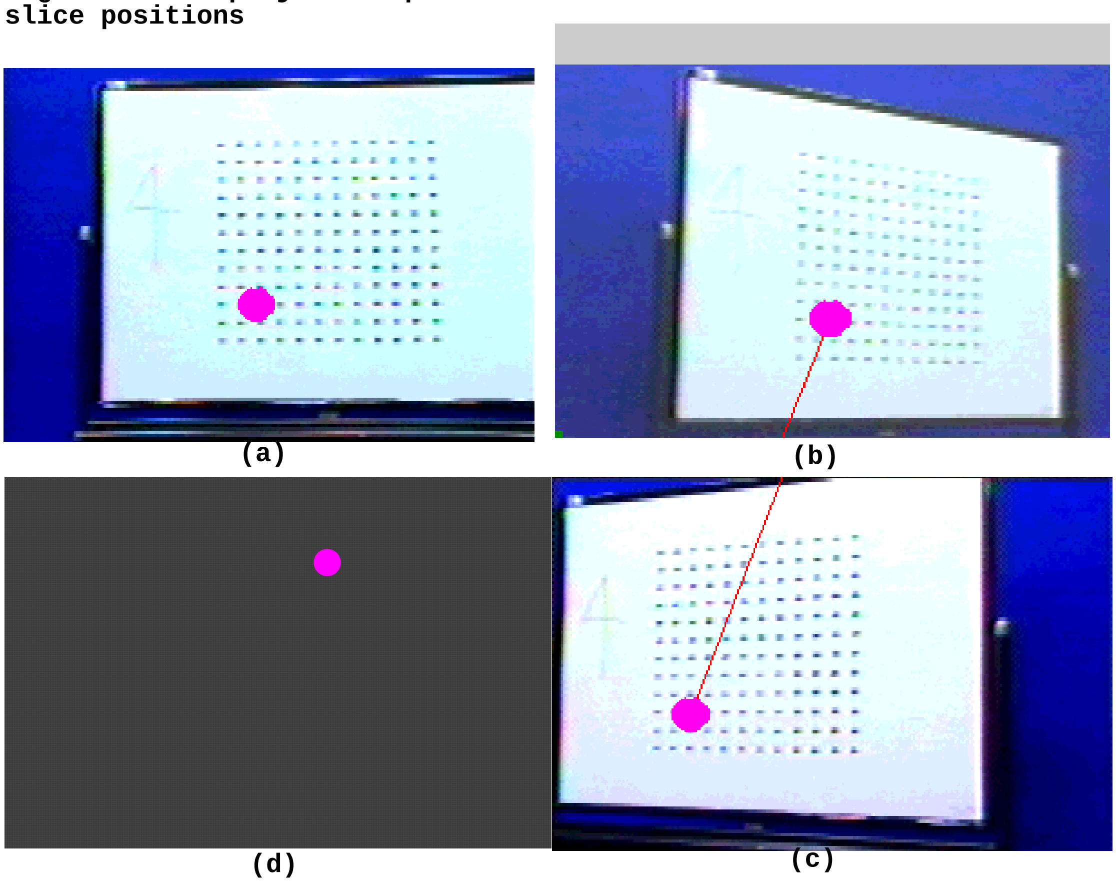
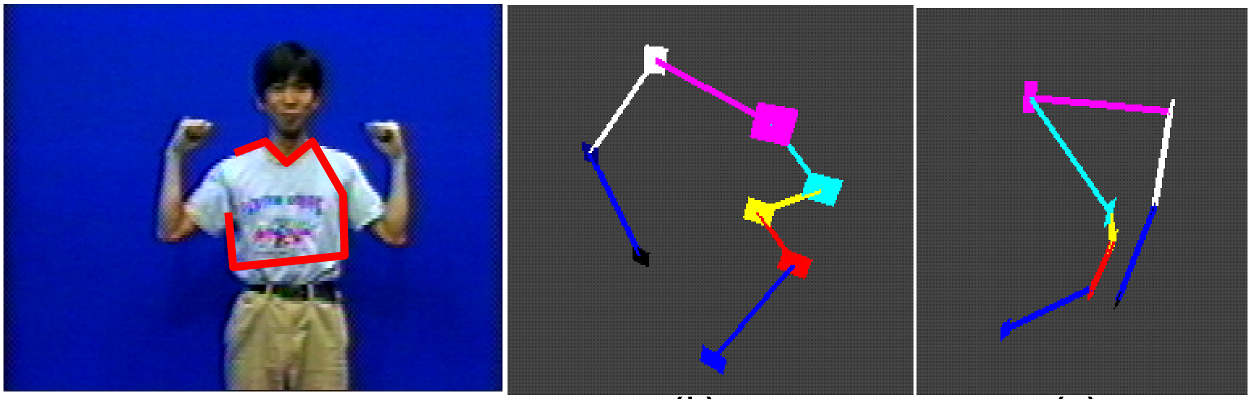
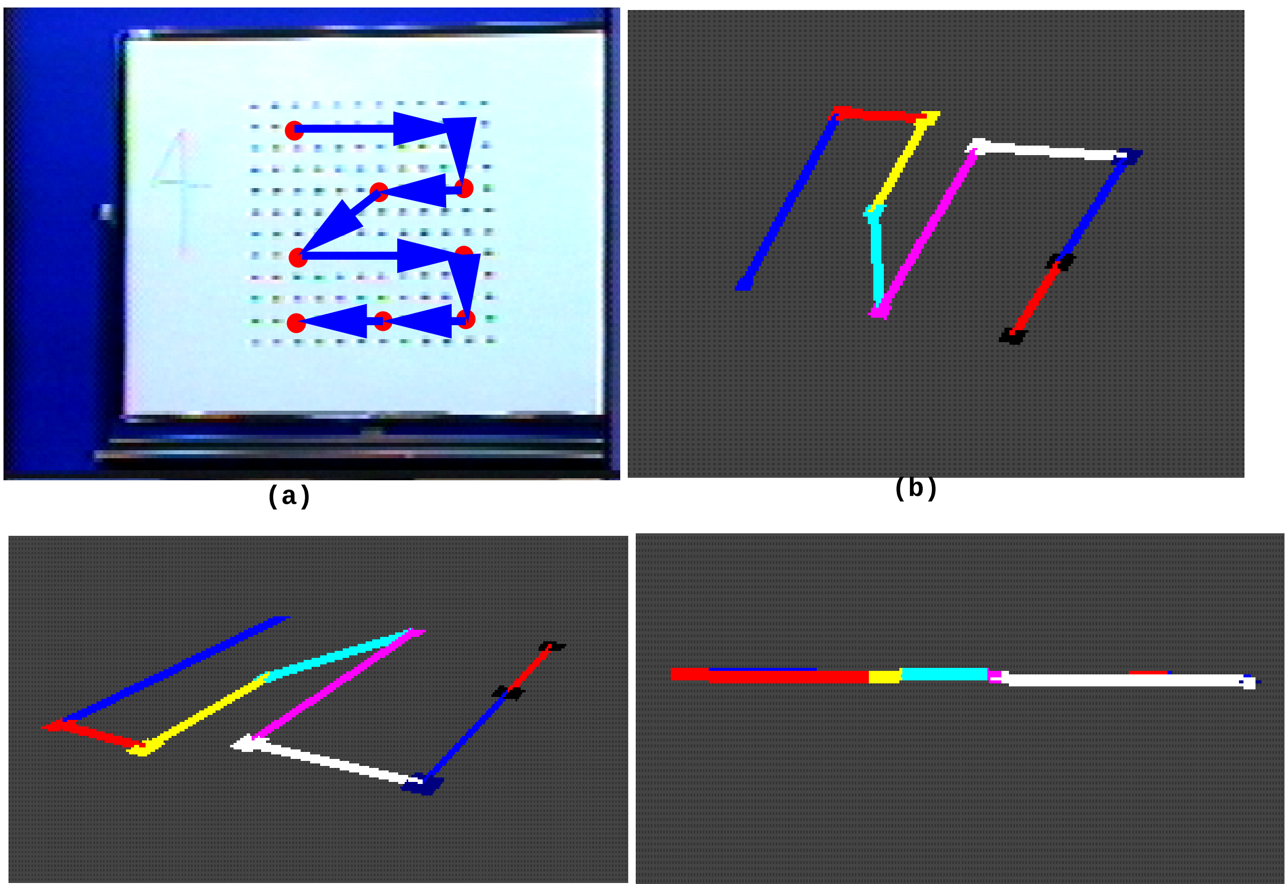


Figure 10: When the same 3Dpoint is specified on all the three camera images (a–c), its precise location (d) can be estimated



(a) (b) (c)

Figure 11: Corresponding pairs are specified using all the three camera images (only one of the three image is shown). The Find_S algorithm is used to obtain the 3D location for every corresponding pair, as shown in (b-c).



(a) (b) (c) (d)

Figure 12: Corresponding points are specified in all the three camera images (only one of them is shown in Figure a). The 3D points estimated by the active space indexing algorithm are shown in Figures b-d.

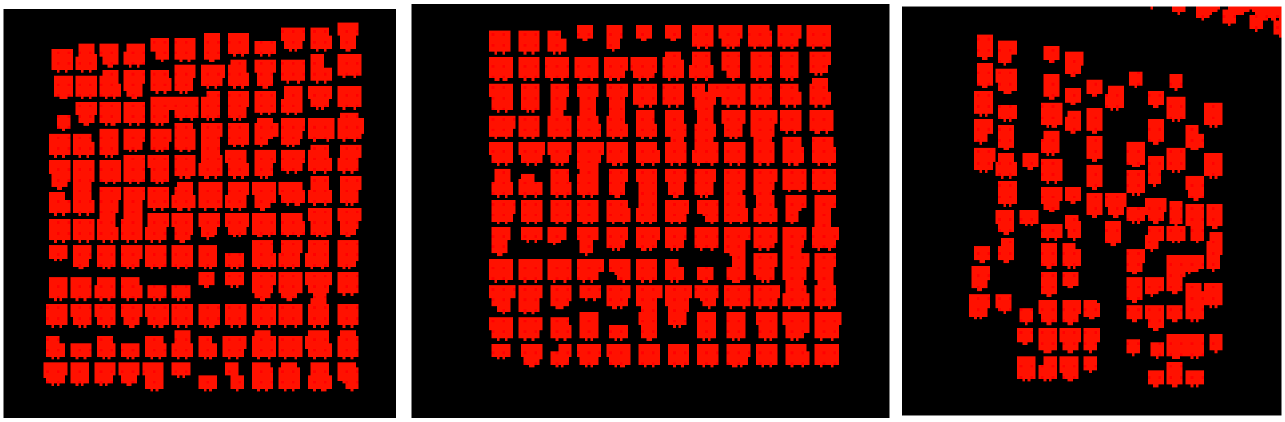


Figure 13: Displaying significant points on a slice using small rectangles. Thresholding is used.

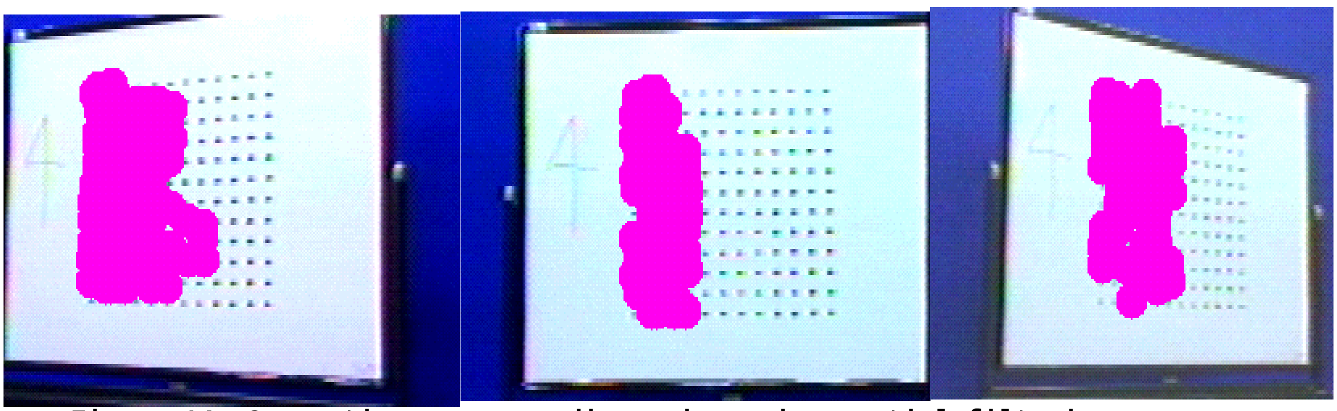


Figure 14: Generating corresponding pairs using spatial filtering

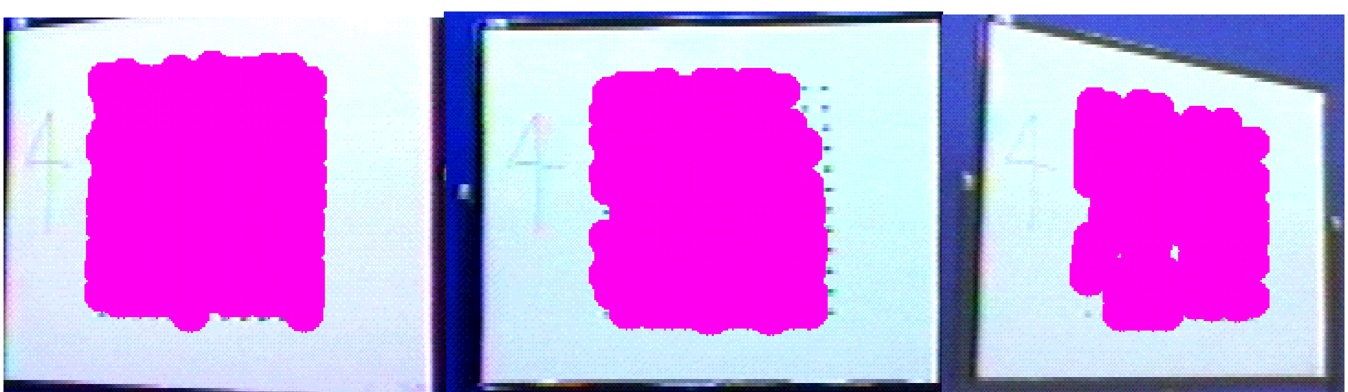


Figure 15: Generating corresponding pairs using spatial filtering

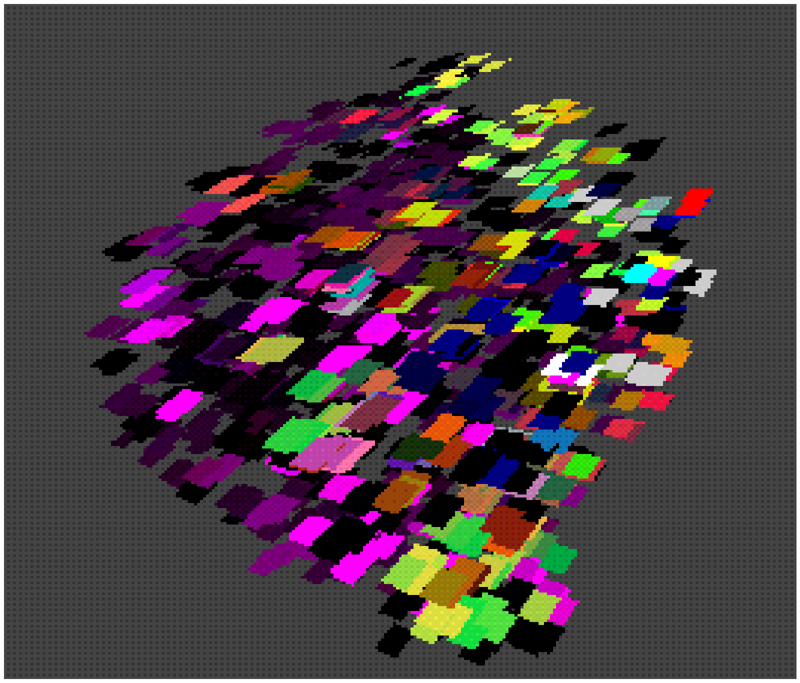


Figure 16: All the corresponding pairs from the eight slices are shown. Each rectangle represents the 3D location of a corresponding pair.