Given an edge (sq1, sq2, r1, r2)

* Make sure the squares aren’t already in the same cluster
  + Call find() on both, make sure that they have different representatives.
* Make sure that gluing that edge won’t create any overlaps in the clusters.
  + Check size of each cluster, let c\_b be the big cluster and c\_s be the small cluster
    - Each representative needs to know the number of squares it is responsible for
  + Determine what rotation will need to be made to c\_s to get it in the proper orientation for concatenation with c\_b
    - Must be able to determine what rotations have already been done to a piece
      * Each node stores a loc\_rot field.
      * When you merge two clusters, update the field of the cluster rep which is made a child of the other rep.
      * When you do a find with path compression, also do rotation and local coordinate compression.
        + return the rep, loc\_rot, and local\_coords of the parent
        + make the node’s parent the rep directly
        + make its loc\_rot be the value it was storing plus whatever its parent returned.
        + Make its local\_coords be its parent’s local\_coords plus the correctly rotated version of its stored local\_coords
  + Determine what the coordinates of the gluing square within c\_s are in the c\_b coordinate system
    - Should be able to look at what edge of the unrotated piece in c\_b we are using, plus what rotations have been done to that piece.
      * Ex: We are using what was originally the left edge of the piece in c\_b, and that piece has been rotated counter-clockwise once as it became part of c\_b. Thus, the piece in c\_s is going to be on top of the piece in c\_b, so we just use that as its
  + Apply the rotation and translation found in the last two steps to the coordinates of each point in c\_s
    - Representative will need to have access to all the coordinates it is responsible for
  + Check if any of the transformed coordinates in c\_s are in c\_b
    - If so, there’s an overlap, don’t glue that edge
  + Merge the clusters, updating all relevant data
    - Make the local\_coords field of the c\_s rep be the translation
    - Make the local\_rot field of the c\_s rep be the rotation
    - Update the c\_b rep to include the new pieces in its piece count and in its list of coords

Once there is only a single cluster remaining, reconstruct

* Look at all the cords stored by the final rep, and get the min x and min y
* Set the local\_coords of the rep to be (-x\_min, -y\_min)
* Call find() starting at every piece. This will update the local\_coords and local\_rot of each piece
* Iterate through each of the pieces, putting the correctly rotated pixels (given by the local\_rot) into the correct spot in the image (given by local\_coords)