The generic RANSAC algorithm, in pseudo code, works as follows:

**input:**  
 data - a set of observations  
 model - a model that can be fitted to data   
 n - the number of data used to fit the model  
 k - the number of iterations performed by the algorithm  
 t - a threshold value for determining when a datum fits a model  
 d - the number of close data values required to assert that a model fits well to data  
**output:**

best\_model - model parameters which best fit the data (or nil if no good model is found)

best\_consensus\_set - data point from which this model has been estimated

best\_error - the error of this model relative to the data

iterations := 0

best\_model := nil

best\_consensus\_set := nil

best\_error := infinity

**while** iterations < k

maybe\_inliers := n randomly selected values from data

maybe\_model := model parameters fitted to maybe\_inliers

consensus\_set := maybe\_inliers

**for** every point in data not in maybe\_inliers

**if** point fits maybe\_model with an error smaller than t

add point to consensus\_set

**if** the number of elements in consensus\_set is > d

*( this implies that we may have found a good model, now test how good it is )*

better\_model := model parameters fitted to all points in consensus\_set

this\_error := a measure of how well better\_model fits these points

**if** this\_error < best\_error

*(we have found a model which is better than any of the previous ones, keep it until a better one is found)*

best\_model := better\_model

best\_consensus\_set := consensus\_set

best\_error := this\_error

iterations++

**return** best\_model, best\_consensus\_set, best\_error

(from Wikipedia)

Please ask the following question:

1. In the beginning of main loop, we randomly select n values from data to fit  
   the **maybe\_model**. How do we decide the value of n? Please explain the reason.
2. In the original RANSAC algorithm, we examine every point in data to generate  
   the **consensus\_set**. This operation is very time consuming and also the bottleneck of the RANSAC algorithm. How can we speed up this part of the algorithm?

Answer:

1. The more sample points we used, the more number of trials we need to find a true set of inliers. The sufficient number of trials grows quickly with the number of sample points used. This provides a strong incentive to use the ***minimum*** number of sample points possible for any given trial, which is how RANSAC is normally used in practice.
2. When the number of measurements is quite large, it may be preferable to **only score a subset of the measurements in an initial round** that selects the most plausible hypotheses for additional scoring and selection. This modification of RANSAC, which can significantly speed up its performance, is called Preemptive RANSAC