ensia
The National Higher School of
Artificial Intelligence
المدرسة الوطنية العليا للذكاء الاصطناعي

Course
« *Computer Vision*»

Sid-Ahmed Berrani

2024-2025

VIII. RANSAC: The Random Sample Consensus Algorithm

Ref.: Fischler and Bolles 1981.

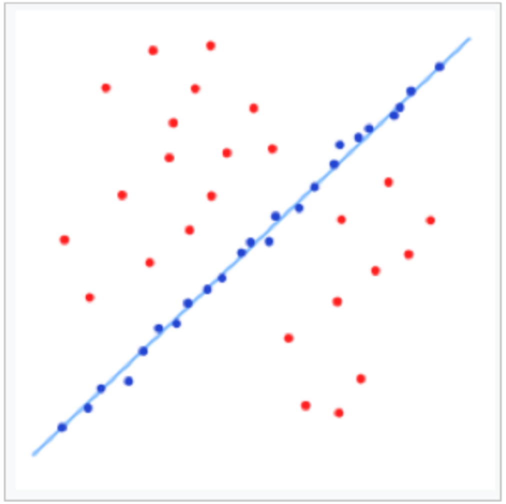
Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, Volume 24, Issue 6. https://dl.acm.org/doi/10.1145/358669.358692

Principle

- A learning technique to estimate parameters of a model by random sampling of observed data.
- Given a dataset whose data elements contain both **inliers** and **outliers**, RANSAC uses a consensus scheme to find the optimal fitting result.

VIII. RANSAC: The Random Sample Consensus Algorithm

RANSAC



- It is a trial-and-error approach.
- Able to deal with high fractions of outliers in the data **(up to 50%)**.

VIII. RANSAC: The Random Sample Consensus Algorithm

RANSAC

- **Key idea:** Find the best partition of inliers/outliers in the data and estimate the model from the inlier subset.
- It the standard/recommended approach for fitting in the presence of outliers.

VIII. RANSAC: The Random Sample Consensus Algorithm

Assumptions

- Data consists of **inliers** (i.e., data whose distribution can be explained by some set of model parameters, though may be subject to noise) and **outliers** which are data that do not fit the model
- Given a (usually small) set of inliers, there exists a procedure which can estimate the parameters of a model that optimally explains or fits this data

For example, given a set of 2 points, we can compute a line model that optimally explains the set.

VIII. RANSAC: The Random Sample Consensus Algorithm

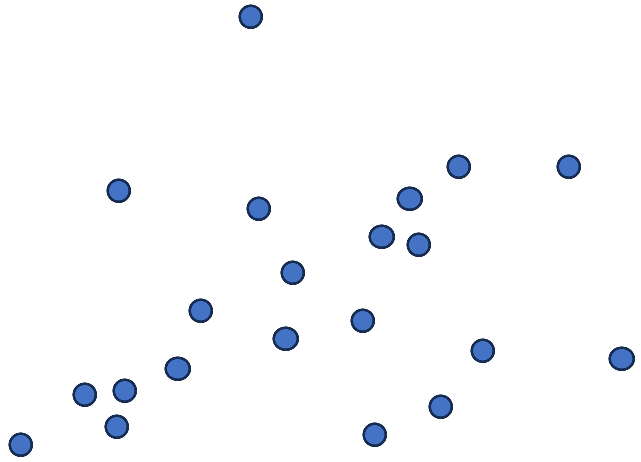
Algorithm

1. Select a random subset of the original data. Call this subset the hypothetical inliers.
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called “consensus set”
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is returned

VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

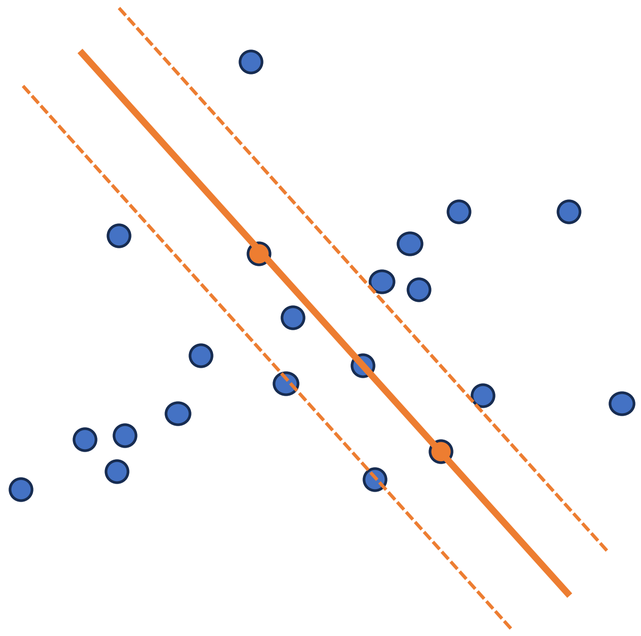
1. Select a random subset of the original data. Call this subset the hypothetical inliers.
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called "consensus set"
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is returned



VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

1. **Select a random subset of the original data. Call this subset the hypothetical inliers.**
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called “consensus set”
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is returned

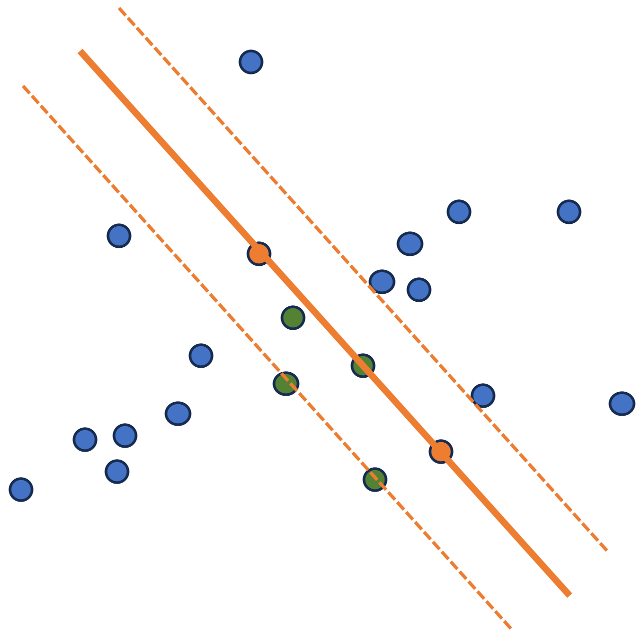


VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

1. **Select a random subset of the original data. Call this subset the hypothetical inliers.**
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called "consensus set"
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is returned

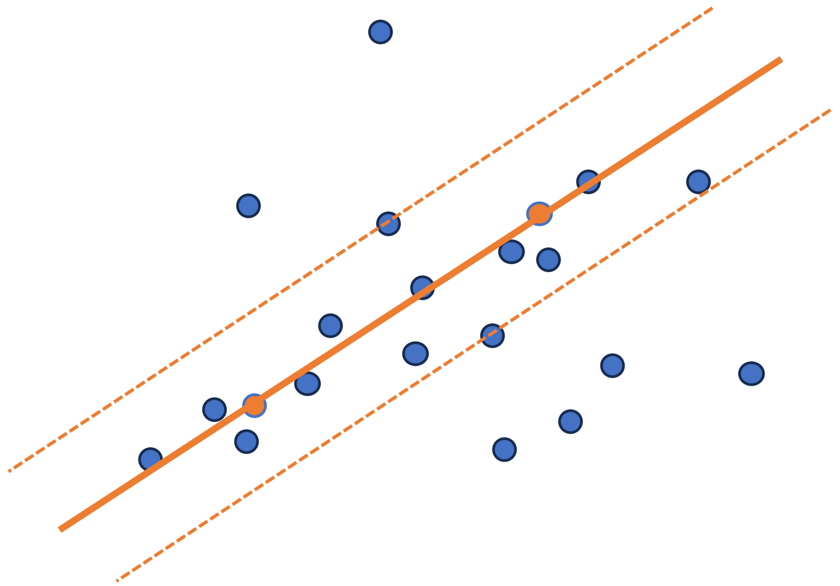
# of inliers = 4



VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

1. Select a random subset of the original data. Call this subset the hypothetical inliers.
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called “consensus set”
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is returned

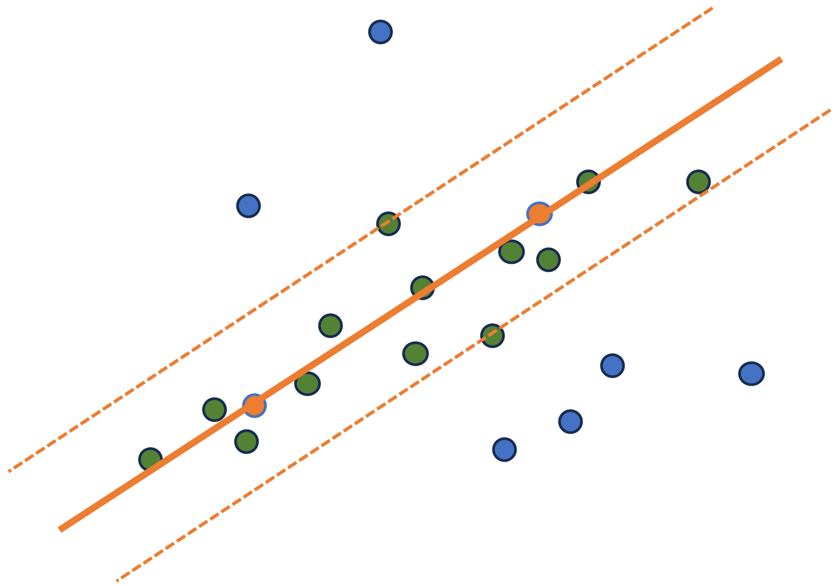


VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

1. **Select a random subset of the original data. Call this subset the hypothetical inliers.**
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called “consensus set”
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is returned

# of inliers = 13



VIII. RANSAC: The Random Sample Consensus Algorithm

Two questions:

1. Which possible final step can be added to the algorithm to improve the quality of the model?

2. How many iteration do we have to perfrom? In other words, how often do we need to try?

VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

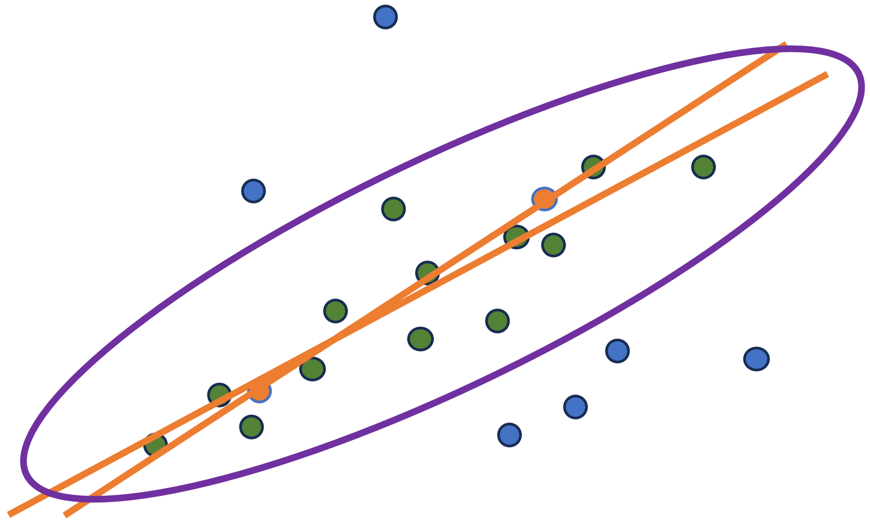
1. Select a random subset of the original data. Call this subset the hypothetical inliers.
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function (a score). The inliers are called “consensus set”
4. Return to step 1 until a predefined number of iterations **N** is reached
5. The model that produced the largest consensus set is selected
6. Fit the model to the set of inliers associated with the selected model.

VIII. RANSAC: The Random Sample Consensus Algorithm

Algorithm

1. Select a random subset of the original data. Call this subset the hypothetical inliers.
2. Fit the model to the hypothetical inliers
3. Test all other data against the model and mark points either as inliers or outliers according to some loss function. The inliers are called “consensus set”
4. Return to step 1 until a predefined number of iterations **N** is reached
5. **The model that produced the largest consensus set is selected**
6. **Fit the model to the set of inliers associated with the selected model.**

# of inliers = 13



VIII. RANSAC: The Random Sample Consensus Algorithm

How many iterations?

s: minimum number of points needed to fit a model

e: Outlier ratio, i.e. probability that a point is an outlier (# outliers / total # of data points).

=> **N:** number of RANSAC interations (i.e. number of trials)

The idea is to choose **N** so that, **with a probability p**, at least one ramdom sample set is free of outliers.

$(1-e)$: Probability to draw one inlier.

$(1-e)^s$: Probability to draw ***s*** inliers.

$1-(1-e)^s$: Probability to fail **once**, i.e. to not draw only inliers.

$(1-(1-e)^s)^N$: Probability to fail **N times**.

$$(1-(1-e)^s)^N = 1-p$$

VIII. RANSAC: The Random Sample Consensus Algorithm

How many iterations?

s: minimum number of points needed to fit a model

e: Outlier ratio, i.e. probability that a point is an outlier (# outliers / total # of data points).

=> **N:** number of RANSAC interations (i.e. number of trials)

The idea is to choose **N** so that, **with a probability p**, at least one ramdom sample set is free of outliers.

$$(1-(1-e)^s)^N = 1-p$$

$$N \log(1-(1-e)^s) = log(1-p)$$

$$\boldsymbol{N = \frac{log(1-p)}{log(1-(1-e)^s)}}$$

VIII. RANSAC: The Random Sample Consensus Algorithm

How many iterations?

$$N = \frac{log(1-p)}{log(1-(1-e)^s)}$$

s: minimum number of points needed to fit a model

e: Outlier ratio.

p: probability of success

=> **N:** number of RANSAC interations (i.e. number of trials)

e	0,1						
	s						
p	**2**	**3**	**4**	**5**	**10**	**15**	**20**
0,1	1	1	1	1	1	1	1
0,5	1	1	1	1	2	4	6
0,75	1	2	2	2	4	7	11
0,9	2	2	3	3	6	10	18
0,95	2	3	3	4	7	13	24
0,99	3	4	5	6	11	20	36
0,999	5	6	7	8	17	30	54
0,9999	6	8	9	11	22	40	72

VIII. RANSAC: The Random Sample Consensus Algorithm

How many iterations?

$$N = \frac{log(1-p)}{log(1-(1-e)^s)}$$

s: minimum number of points needed to fit a model

e: Outlier ratio.

p: probability of success

=> **N:** number of RANSAC interations (i.e. number of trials)

e	0,3						
	s						
p	**2**	**3**	**4**	**5**	**10**	**15**	**20**
0,1	1	1	1	1	4	23	132
0,5	2	2	3	4	25	146	869
0,75	3	4	6	8	49	292	1 737
0,9	4	6	9	13	81	484	2 885
0,95	5	8	11	17	105	630	3 753
0,99	7	11	17	26	161	968	5 770
0,999	11	17	26	38	242	1 452	8 654
0,9999	14	22	34	51	322	1 936	11 539

VIII. RANSAC: The Random Sample Consensus Algorithm

How many iterations?

$$N = \frac{log(1-p)}{log(1-(1-e)^s)}$$

s: minimum number of points needed to fit a model

e: Outlier ratio.

p: probability of success

=> **N:** number of RANSAC interations (i.e. number of trials)

e	0,5						
	s						
p	**2**	**3**	**4**	**5**	**10**	**15**	**20**
0,1	1	1	2	4	108	3 453	110 479
0,5	3	6	11	22	710	22 713	726 818
0,75	5	11	22	44	1 419	45 426	1 453 635
0,9	9	18	36	73	2 357	75 450	2 414 435
0,95	11	23	47	95	3 067	98 163	3 141 252
0,99	17	35	72	146	4 714	150 900	4 828 869
0,999	25	52	108	218	7 071	226 350	7 243 303
0,9999	33	69	143	291	9 427	301 800	9 657 738

VIII. RANSAC: The Random Sample Consensus Algorithm

The minimum number is points *s* matters

- If for a problem, we have two algorithm that require different values of ***s***:

  s can be used as a decision criterion

  A smaller value of ***s*** => A more efficient RANSAC procedure.

RANSAC – Pros

- Robust to deal with outliers
- Works well with $s \in [1..10]$ (depending on *e*)
- Easy to understand and to implement

RANSAC – Cons

- Computational times grows quickly with *e* and *s*.
- Not good for getting multiple fits.

VIII. RANSAC: The Random Sample Consensus Algorithm

Applications

- Robust Line and Plane Fitting

  Fit lines in 2D, planes in 3D, even when many points are outliers.

- Homography Estimation

  Find projective transformations between two images.

  Key in panorama stitching, augmented reality, image registration.

- Model Fitting in Object Recognition

  Fit shapes (lines, circles, ellipses, 3D models) to detected features.

  Recognize partially occluded or noisy objects.

- Pose Estimation (PnP problem)

  Estimate a camera's position and orientation from known 3D-2D point correspondences.

  Used in robotics, AR, self-driving cars.

- …