RANSAC With Likelihood Sampling

Xiangyu Kong
University of Toronto
1002109620 kongxi16

Zun Cao
University of Toronto
1003871250 caozun

Abstract

RANSAC (RAndom SAmple Consensus) has been a popular algorithm in computer vision for fitting a model to data points containing outliers [2]. Traditional RANSAC algorithm uses random sampling to choose data points, and use these data points to generate models [4]. Instead of sampling randomly, we explore the possibility of sampling based on the likelihoods of the data point to be inliers. A previously developed form of RANSAC, BAYSAC [1], is used in the experiment. Experiments were conducted on the noisy circle finding problem [3] in order to demonstrate the differences and similarities between BAYSAC and RANSAC.

1. Introduction

1.1. RANSAC

RANSAC (RAndom SAmple Consensus) is mainly used in computer vision for fitting a model to data points containing outliers [2]. In general, RANSAC algorithm follows the algorithm given in Algorithm 1:

Algorithm 1 RANSAC($num\_iter, threshold$)

$t = 0, \mathbb{I} = \{\}$
while $t < num\_iter$ and $\mathbb{I}.size() < desired\_size$ **do**
  Randomly Sample dataset H_i of size $(n + 1)$.
  Fit a polynomial $model$ of degree n.
  if $\forall j \in H_i, dist(model, j) < threshold$ **then**
    $\mathbb{I} = \mathbb{I} \cup H_i$.
  end if
  $t++$
end while
return the fitted $model$ using all points in $\mathbb{I}$.

1.2. Problem Definition

The problem of noisy circle finding problem is defined as follows. Given two input arrays of pixel locations where $X = [x_1, x_2, \ldots, x_n]$, $Y = [y_1, y_2, \ldots, y_n]$, $n \geq 3$, find a best-fit circle with center (x, y), and its radius r [3]. An example of the problem is shown in Figure 1

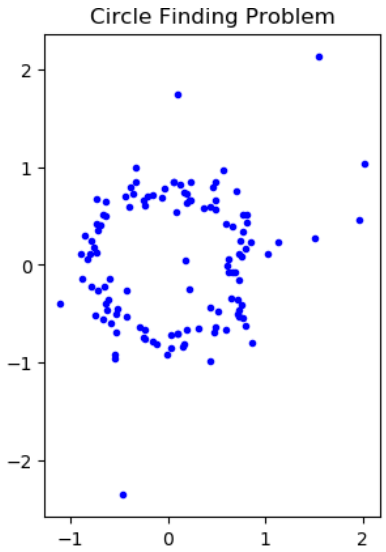


Figure 1: Example of Circle Finding Problem

2. Related Work

2.1. Noisy Circle Finding Problem

The noisy circle finding problem has been a classic exercise for computer vision students and an ongoing research topic for scholars. There are many approaches to solving this problem, including RANSAC, Patch Matching, Neural Network, etc.

For our experiment, we adopted and extended on an existing RANSAC circle detection project [5]. The original project uses RANSAC with a twist. Instead of counting the number of inliers, the author chooses the model that has the minimum distance to all data points. We modified this part back to counting the number of inliers, and extended the class to implement the BAYSAC algorithm described in Section 2.2.

2.2. BAYSAC

The BAYSAC algorithm [1] first assign all points' likelihoods to be inliers to 0.5. For each iteration, it updates the points' likelihoods using Equation (1), where $P_t(i \in \mathbb{I})$ indicates the probability of point i being in the inliers set $\mathbb{I}$ at iteration t; $P(H_t \subseteq \mathbb{I})$ indicates the probability of the sample H at iteration t being a subset of the inlier set $\mathbb{I}$. I.e. for all points in the sample $j \in H_t$, $j \in \mathbb{I}$.

$$P_t(i \in \mathbb{I}) = \begin{cases} \dfrac{P_{t-1}(i \in \mathbb{I})P(H_t \not\subseteq \mathbb{I} \,|\, i \in \mathbb{I})}{P(H_t \not\subseteq \mathbb{I})} & i \in H_t \\ P_{t-1}(i \in \mathbb{I}) & i \notin H_t \end{cases} \tag{1}$$

To calculate $P(H_t \not\subseteq \mathbb{I} \,|\, i \in \mathbb{I})$ and $P(H_t \not\subseteq \mathbb{I})$, Equation (2) and Equation (3) are used.

$$\begin{aligned} P(H_t \not\subseteq \mathbb{I}) &= 1 - P(H_t \subseteq \mathbb{I}) \\ &= 1 - \prod_{j \in H_t} P_{t-1}(j \in \mathbb{I}) \end{aligned} \tag{2}$$

$$\begin{aligned} P(H_t \not\subseteq \mathbb{I} \,|\, i \in \mathbb{I}) &= 1 - P(H_t \subseteq \mathbb{I} \,|\, i \in \mathbb{I}) \\ &= 1 - \prod_{j \in H_t; j \neq i} P_{t-1}(j \in \mathbb{I}) \\ &= 1 - \frac{P(H_t \subseteq \mathbb{I})}{P_{t-1}(i \in \mathbb{I})} \end{aligned} \tag{3}$$

Finally, combining Equations (2, 3) with Equation (1), we derive Equation (4)

$$P_t(i \in \mathbb{I}) = \begin{cases} \dfrac{P_{t-1}(i \in \mathbb{I}) - P(H_t \subseteq \mathbb{I})}{1 - P(H_t \subseteq \mathbb{I})} & i \in H_t \\ P_{t-1}(i \in \mathbb{I}) & i \notin H_t \end{cases} \tag{4}$$

3. Experiment

3.1. Algorithm

Combining the two related work described in the Section 2, the final BAYSAC algorithm can be written as Algorithm 2, where $dist(model, j)$ is the L2 distance defined by Equation 5.

$$dist(x_c, y_c, r_c, x_j, y_j) = |\sqrt{(x_j - x_c)^2 + (y_j - y_c)^2} - r_c| \tag{5}$$

Algorithm 2 BAYSAC(num_iter, $threshold$)

$t = 0, \mathbb{I} = \{\}$
while $t < num\_iter$ and $\mathbb{I}.size() < desired\_size$ **do**
  Sample H_t of $(n+1)$ points ordered by highest $P_{t-1}(i \in \mathbb{I})$
  Fit a polynomial $model$ of degree n.
  if $\forall j \in H_t, dist(model, j) < threshold$ **then**
    $\mathbb{I} = \mathbb{I} \cup H_t$.
  end if
  update all points' $P_t(i \in \mathbb{I})$ base on Equation 4
  $t++$
end while
return the fitted $model$ using all points in $\mathbb{I}$.

3.2. Experimental Design

The parameters for the algorithms are the total number of datasets ($total\_num$), the circle to noise data point ratio ($ratio$), the circle points noise variance ($circle\_noise$) and the noisy points variance ($noisy\_noise$).

For each parameter setting, 10 iterations were run. For each iteration, two algorithms are run on the same set of generated data points. The analytic data are collected for each iteration and then averaged and plotted.

The analytic data collected after the experiment are percentage of inliers that were actual circle data (accuracy), runtime, the distance from the fitted model to the inliers, and the differences between the fitted circle's centers and radiuses using two algorithms.

3.3. Results

Baseline Group

The default parameters we set were $total\_num = 1000$, $ratio = 0.8$, $circle\_noise = 0.1$, $noisy\_noise = 1.0$. Due to the page constraint, only three pairs of results of the two algorithms are shown in Figure 2. See project repository [6] for complete result. Observe that with the same dataset, RANSAC and BAYSAC produce very similar results with slight deviation.

Total Number of Data Points

This part of the experiment tests the effect of the total number of data points. The number of data varies from 100 to 10100. For each epoch, the number of data is increased by 1000. The accuracy, runtime, distance to inliers and difference between two algorithms' fitted circles' centers and radiuses for each epoch is then plotted and shown in Figure 3.

Circle to Noise Ratio

This part of the experiment tests the circle to noise data point ratio. The ratio varies from 0.0 (only noise) to 1.0 (only circle). For each epoch, the ratio is increased by 0.1. The accuracy, runtime, distance to inliers and difference be-

tween two algorithms' fitted circles' centers and radiuses for each epoch is then plotted and shown in Figure 4.

Number of Iterations

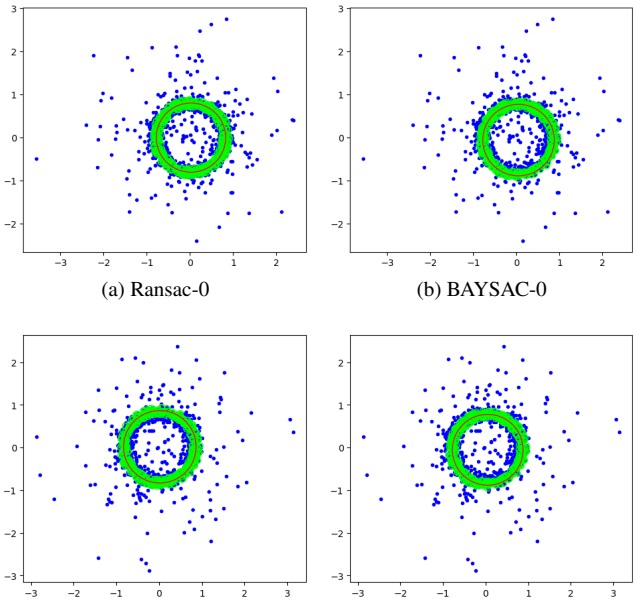
This part of the experiment tests the number of iterations. The number of iterations varies from 2 to 40. For each epoch, the number of iterations is increased by 2. The accuracy, runtime, distance to inliers and difference between two algorithms' fitted circles' centers and radiuses for each epoch is then plotted and shown in Figure 5.

Circle Noise

This part of the experiment tests the circle noise variance. The variance of circle noise varies from 0.0 to 1.0. For each epoch, the variance is increased by 0.2. The accuracy, runtime, distance to inliers and difference between two algorithms' fitted circles' centers and radiuses for each epoch is then plotted and shown in Figure 6.

Noisy Noise

This part of the experiment tests the Noisy points' noise variance. The variance varies from 0.0 to 2.0. For each epoch, the variance is increased by 0.5. The accuracy, runtime, distance to inliers and difference between two algorithms' fitted circles' centers and radiuses for each epoch is then plotted and shown in Figure 7.



(a) Ransac-0 (b) BAYSAC-0

(c) Ransac-1 (d) BAYSAC-1

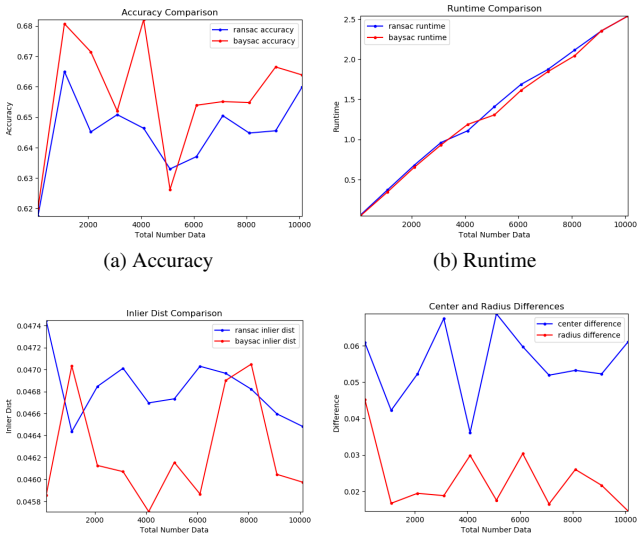
Figure 2: Baseline Results

4. Conclusions

Base on the results shown in Section 3.3, we can see that under the problem of noisy circle detection, RANSAC and BAYSAC performs almost equally well. The accuracies and runtimes for two the two algorithms are very similar. Note that for Figure 7a, the y-axis scale is enlarged because the accuracies for RANSAC and BAYSAC are very similar.

By looking at the Center and Radius Differences, the results of RANSAC and BAYSAC converges as the number of iterations increases, circle to noise ratio increases (there are more circle points than random noise given the same number of total data points) and circle noise decreases.

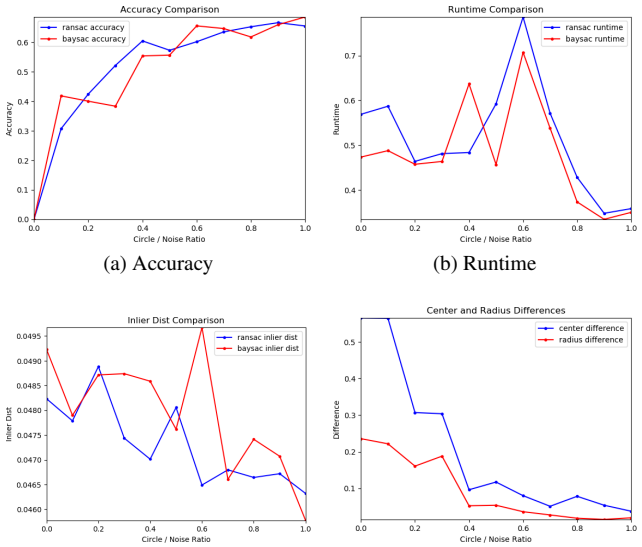
In the original BAYSAC paper [1], the author claimed



(a) Accuracy (b) Runtime

(c) Inlier Distance (d) Center and Radius Difference

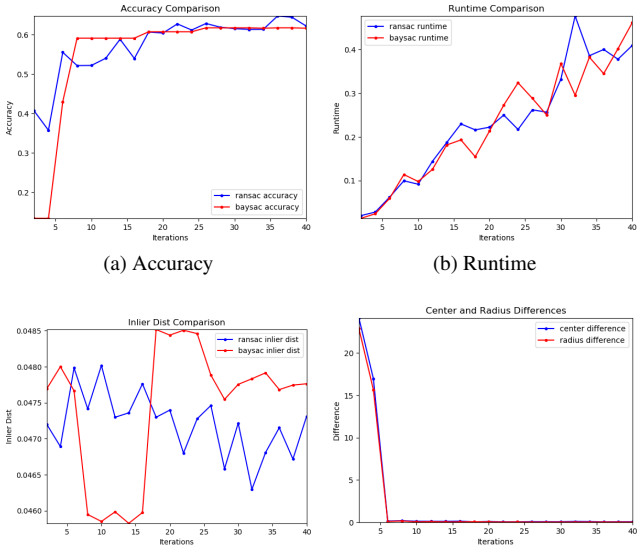
Figure 3: Total Number of Data



(a) Accuracy (b) Runtime

(c) Inlier Distance (d) Center and Radius Difference

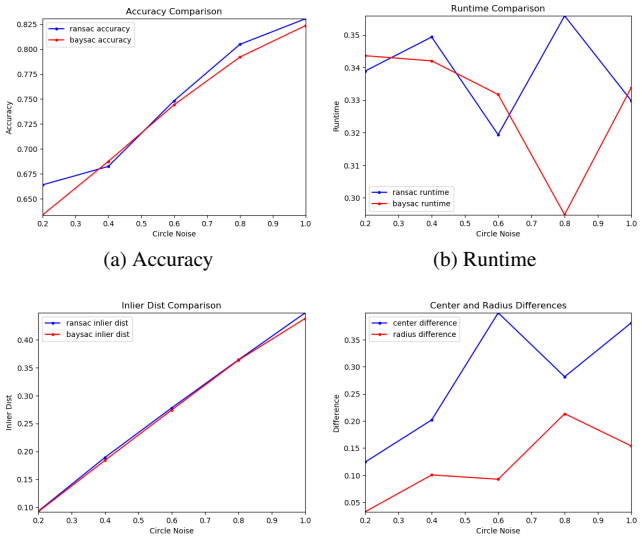
Figure 4: Circle to Noise Ratio



(a) Accuracy (b) Runtime

(c) Inlier Distance (d) Center and Radius Difference

Figure 5: Number of Iterations

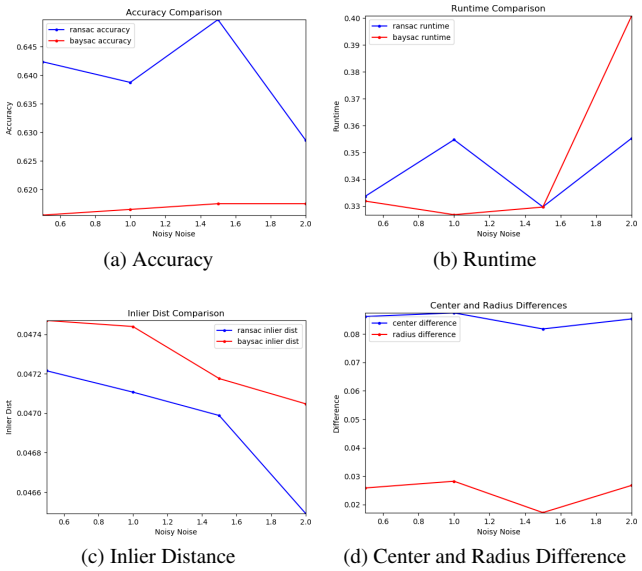


(a) Accuracy (b) Runtime

(c) Inlier Distance (d) Center and Radius Difference

Figure 6: Circle Noise

that BAYSAC could potentially help reduce the failure rate by up to 78% compared to vanilla RANSAC. However, we were not able to show this through our experiment. One potential reason is that the noisy circle detection problem is at a very low dimension, and circle is a very low degree polynomial ($n = 2$). In the future, one possible improvement to this experiment is to compare RANSAC and BAYSAC results for solving a higher dimension problem.



(a) Accuracy (b) Runtime

(c) Inlier Distance (d) Center and Radius Difference

Figure 7: Noisy Noise

References

[1] Tom Botterill, Steven Mills, and Richard D Green. New conditional sampling strategies for speeded-up ransac. In *BMVC*, pages 1–11. Citeseer, 2009.

[2] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[3] Yawen Ma. 2019 fall midterm, 2019. Question 4 in University of Toronto CSC320 Fall 2019 Midterm.

[4] Yawen Ma. Local analysis of image patches, 2019. Lecture 4 for University of Toronto CSC320 Winter 2019.

[5] Bae Seong-hyun. Ransac-circle-python, Jan. 2019. GitHub repository.

[6] Zun Cao Xiangyu Kong. Csc320-final-project, Mar. 2020. GitHub repository.