Lecture05a: Particle Filters

CS540 2/13/18

Announcements

On-campus students:
- Make sure I am wearing the microphone

Off-campus students
- Keep up. Expectations are the same as for on-campus students

All students:
- How are the projects coming?
  - Dejan has his doubts…
- Thursday will be project presentations
  - Send me Powerpoint or pdf files by Wednesday night

Simple Genetic Algorithm

1. Create initial population P with N elements x^i
2. For I = 1 to N do
   1. Calculate $f(x^i)$
3. While (not termination criterion)
   1. Create M with N individuals selected from P (resampling based on f)
   2. Index = 1
   3. Repeat
      1. If random(0,1) ≤ p recombine x^i and x^{i+1} and add to P'
      2. Else copy x^i and x^{i+1} to P'
      3. Index += 2
   4. Until index > N
   5. For i =1 to N
      for j = 1 to |x|
      if random(0,1) ≤ p then mutate(x^i_j) in P'
   6. Calculate $f(x^i)$ for P'
   7. P = P'

GAs are great for static problems

Problems like
- Satisfiability
- Traveling Salesman
- *Static* path planning

But what if the problem changes over time?
- What if the evaluation function is a function of state & time?
- So the optimal answer at time *t* is no longer optimal at *t+1*?

OK, but doesn't really happen… does it?

Tracking

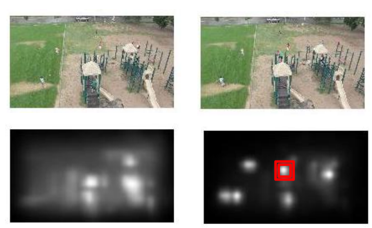
Task:

Track children on playground

Measure their activity

Part of childhood obesity & playground equipment effectiveness study

Motion Data



Motion Data:
1. Model background
2. Label every pixel by probability of being foreground
3. Fitness function
   (a) target pixels should be foreground
   (b) target should be surrounded by background

Tracking and GAs

There is a tight connection between *Genetic Algorithms* and *Particle Filters* and *Kalman Filters*

Let's look into it…

Tracking as a Dynamic System

Targets move over time

- Partially observable state: $\begin{bmatrix} x \\ y \\ \Delta x \\ \Delta y \\ w \\ h \end{bmatrix}$
- Motion model predicts next state based on (x, y, *Δ*x, *Δ*y)
- Fitness based on (x, y, w, h)

Observations

- Noisy
- Subject to occlusions & FOV

Multiple Targets

Modify GAs for Tracking (step 1)

1. Create initial population P with N elements x^i
2. For I = 1 to N do
   1. Calculate f(x^i,t)
3. While (not termination criterion)
   1. Create M with N individuals selected from P (resampling based on f)
   2. Index = 1
   3. Repeat
      1. If random(0,1) ≤ p recombine x^i and x^{i+1} and add to P'
      2. Else copy x^i and x^{i+1} to P'
      3. Index += 2
   4. Until index > N
   5. For i =1 to N
      for j = 1 to |x|
      if random(0,1) ≤ p then mutate(x^i_j) in P'
   6. Calculate f(x^i,t) for P'
   7. P = P'

Note: *t* is time, not iteration

Modify GAs for Tracking (step 2)

1. Create initial population P with N elements x^i
2. For I = 1 to N do
   1. Calculate f(x^i,t)
3. While (not termination criterion)
   1. Create M with N individuals selected from P (resampling based on f)
   2. ~~Index = 1~~
   3. ~~Repeat~~
      1. ~~If random(0,1) ≤ p recombine xi and xi+1 and add to P'~~
      2. ~~Else copy xi and xi+1 to P'~~
      3. ~~Index += 2~~
   4. ~~Until index > N~~
   5. For i =1 to N
      for j = 1 to |x|
      if random(0,1) ≤ p then mutate(x^i_j) in P'
   6. Calculate f(x^i,t) for P'
   7. P = P'

Particle Filters are GAs with selection & mutation, but not crossover.

Modify GAs for Tracking (step 3)

1. Create initial population P with N elements x^i
2. For I = 1 to N do
   1. Calculate f(x^i,t)
3. While (not termination criterion)
   1. Create M with N individuals by
      1. Selecting samples at time t according to their fitness
      2. Forward predicting the sample to time t+1
   2. For i =1 to N
      for j = 1 to |x|
      if random(0,1) ≤ p then mutate(x^i_j) in P'
   3. Calculate f(x^i,t) for P'
   4. P = P'

Particle Filters are GAs with dynamic models for prediction.

Extensions

It is possible to add a little local search

- Fit each track by optimizing the fitness function in a small search window on each step
  - Search in x, y, w &h
  - Reset to local optimum
- Tends to increase redundancy, hurt diversity
- Increases cost (time)

It is possible to adapt the model

- Reset *Δ*x and *Δ*y to average motion over last N time steps

Notes

Particle Filters rely on redundancy
- Likelihood of a track propagating from *t* to *t+1* is
  - Based on fitness
  - Based on number of similar tracks (tracking same object)
- Otherwise noise or temporary occlusion could end track

At termination
- Select tracks based on probability over time
  - Not final value of fitness function
- Still need to threshold tracks
  - Since there is no *single* answer

But we haven't really talked about forward prediction or probabilities. From Particle filters to Kalman Filters!