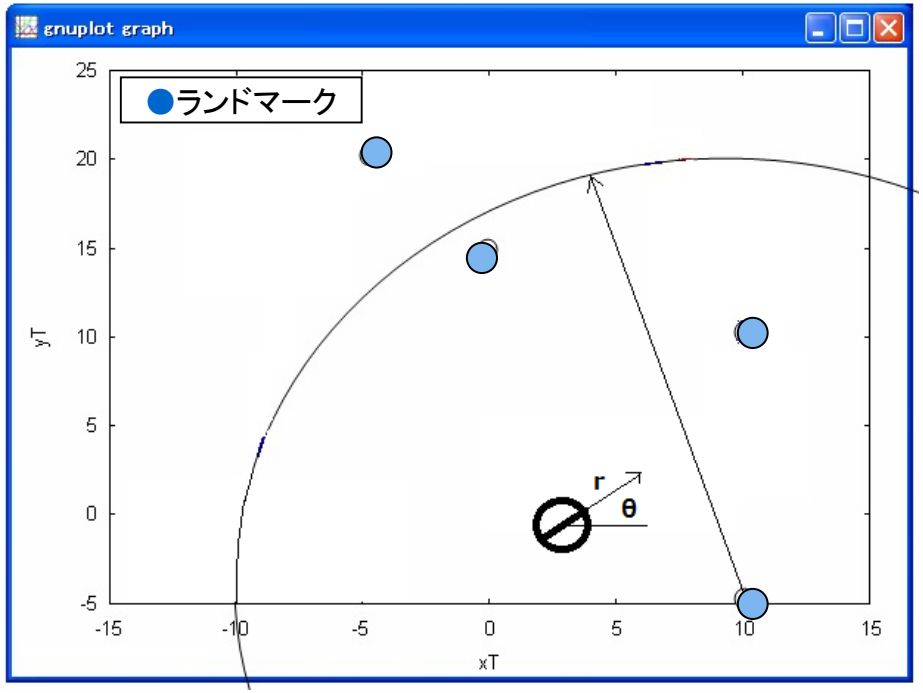
粒子フィルターによる
自動運転

2017/04/08

mabonki0725

確実なものが無い世界の制御

- ロボットはモータとハンドルでΔTあたり走行距離rと走行角θで走行する
  - モータトルクに誤差があり、走行距離rに平均±0.1の誤差がある。
  - ハンドル感度に誤差があり走行角θに平均±3°の誤差がある。
- ロボットは4箇所のランドマークからの信号で距離d_1～d_4をΔT毎に観測
  - 観測距離d_1～d_4には平均±0.1の**誤差**がある。
- ランドマークからの信号到達距離には限界があり、**超えると観測できない**

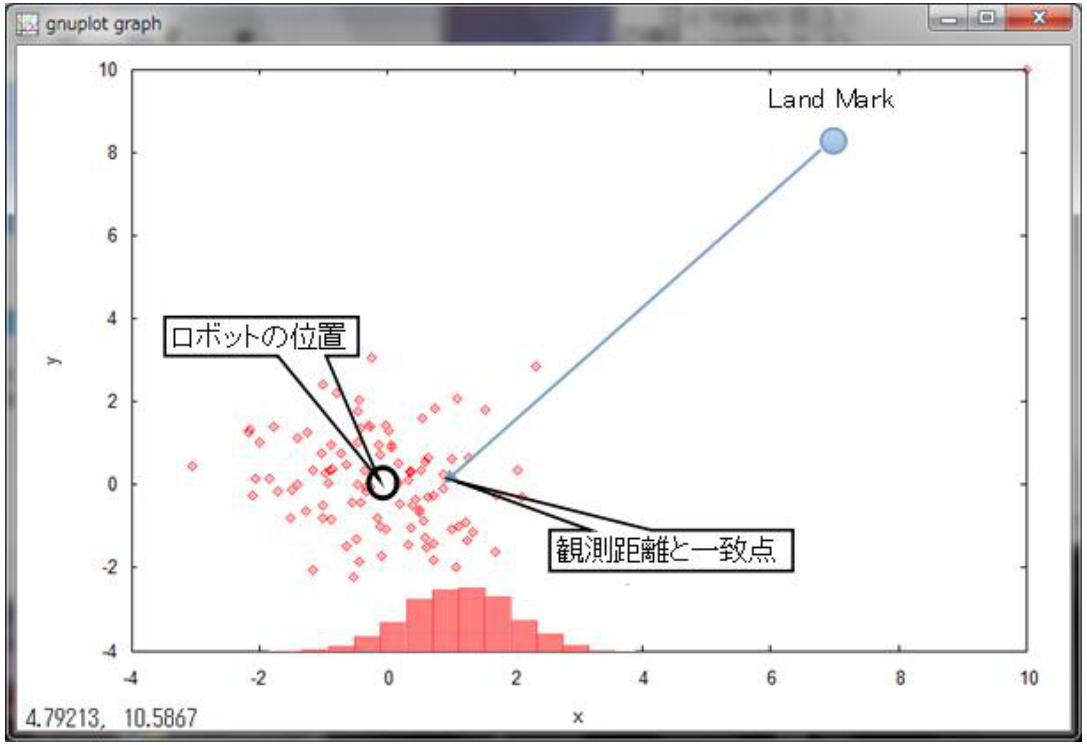


粒子フィルター による位置推定

- ロボットの自己位置をΔT毎に粒子フィルター（100個）を飛ばして各粒子の尤度pを推定
  - 前の位置から移動(r, θ)した点回りで100個粒子をばら撒くx[i] i=1～100
  - 各粒子とランドマークとの観測距離から尤度p[i]を計算

  p[i] = N(x[i] / d1,～,d4)

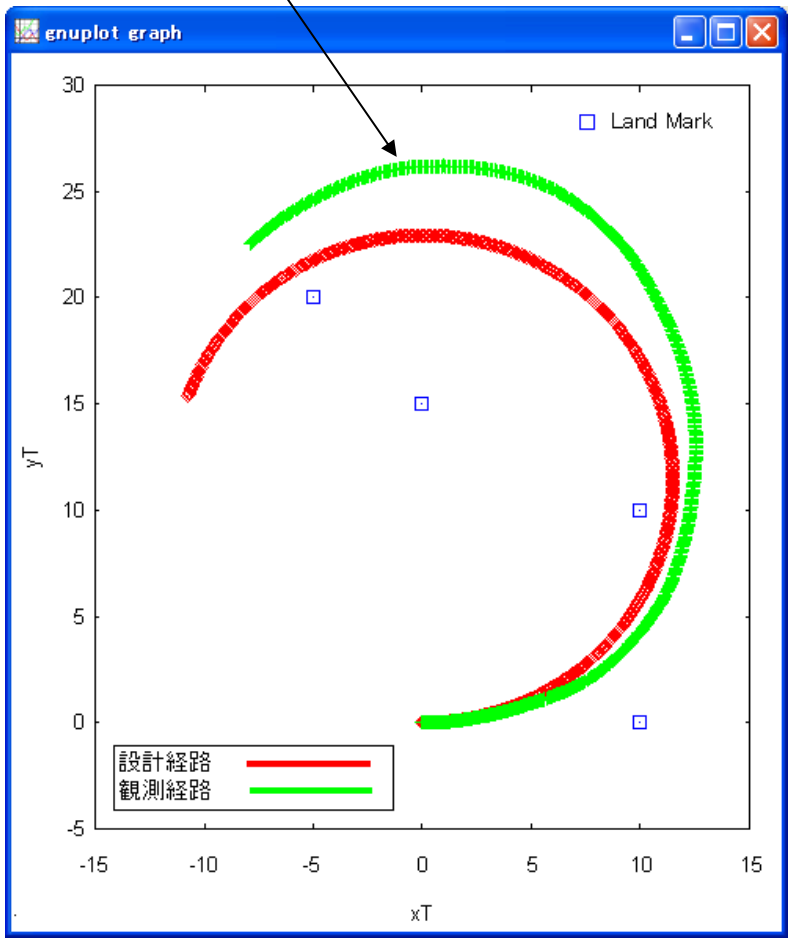
  i=1～100
- 自己位置の推定
  - X= Σ x[i] * p[i]
- 粒子の大きさを調整
  - Resampling

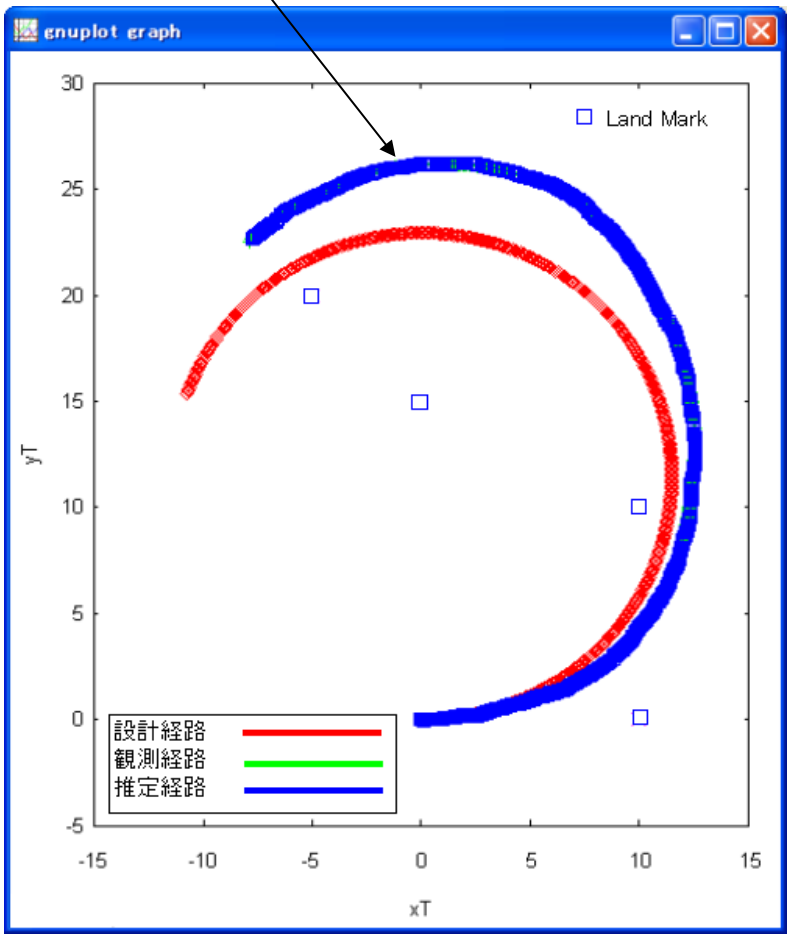


粒子位置と尤度の関係

計算結果(自己位置推定)

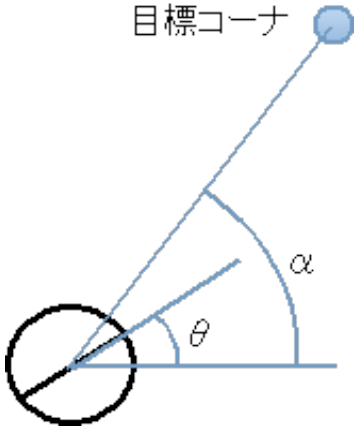
- 観測経路と粒子フィルターによる推定経路は一致している



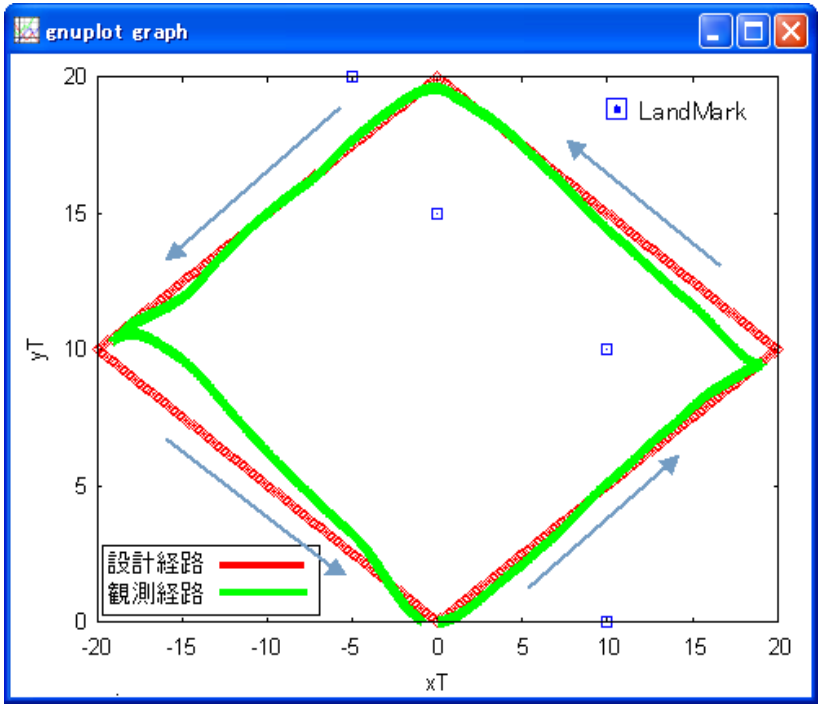


計算結果(自動運転)

- 4コーナを回る自動運転
  1. 反時計回りに4点のコーナを指定
  2. ロボットは自己位置と走行方向を推定しながらコーナに向かう
     LandMarkからの距離で尤度計算
  3. 目標とするコーナとの角度αと走行角度θより角度を微調整(10%)
     調整角度=(θ-α)/10

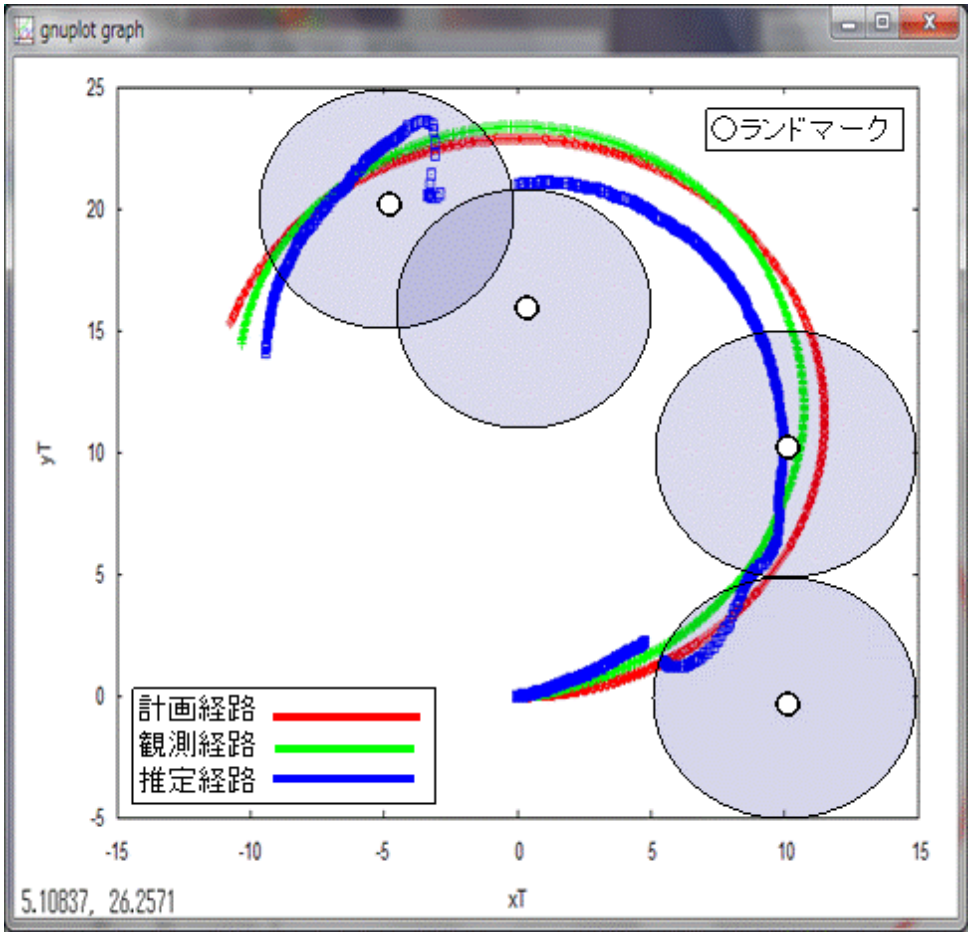


  4. コーナの近傍に達すると次のコーナに向かう



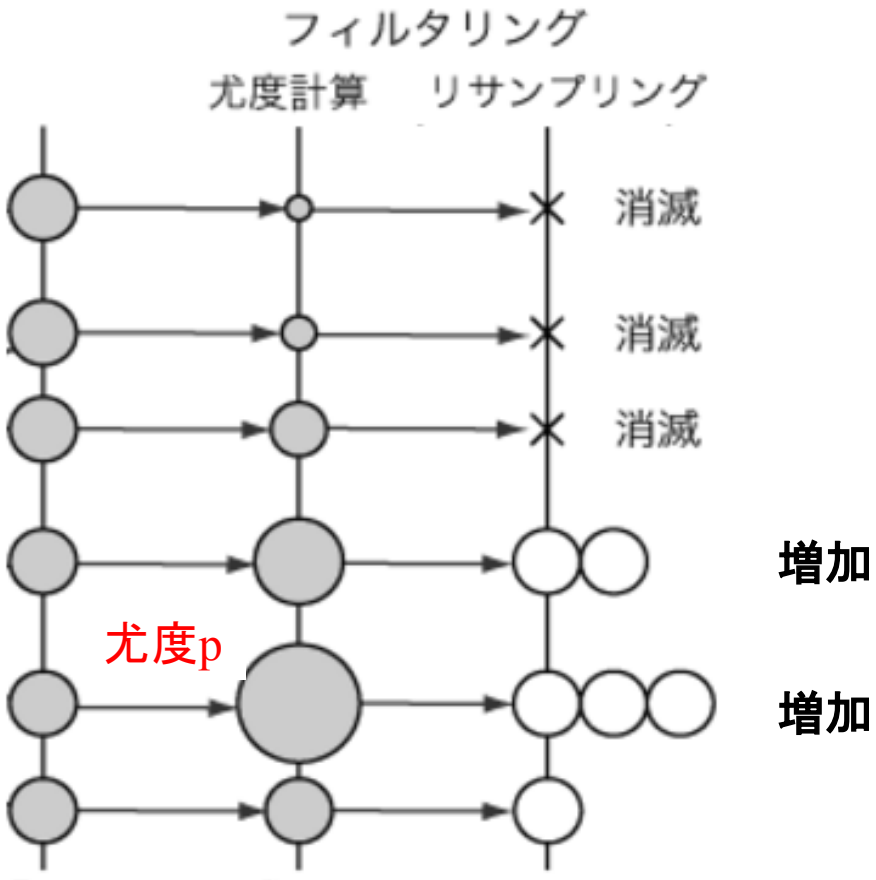
計算結果（観測範囲縮小）

- 観測可能範囲から逸脱すると推定誤差が拡大
  - 各ランドマークからの観測範囲20→5に縮小



Resampling

大きな尤度の粒子を分割して消滅した粒子を補う



自動運転はベイズ統計モデル

- ベイズ統計とは、観測から実態（要因）を予測するモデル
  - 実態の挙動zが分かればこの観察データxは予測し易い。しかし観測データxは得られるが実態zは不明な事が多い
  - 観測データxから実態の挙動zの条件付き確率p(z|x)を求めたい
  - p(z|x)はベイズの定理を展開して計算できる
  - p(z|x)が求められれば観測データxの場合、最も期待できるzが分かる
  - 今回の場合は
    - zi:自動車からばら撒かれた粒子の位置　i=1～100
    - xi：ランドマークから粒子までの観測距離データ
    - 自動車の位置は粒子の平均位置と推定= Σ 粒子の位置 × p(zi|xi)

推定し易い

ベイズの定理

$$p(z \mid x) = \frac{p(x \mid z)p(z)}{p(x)}$$

z：実態の挙動

x：観察データ

粒子フィルターのプログラム

```
for(i=0; i <nSteps; i++) {
  time = time + dt;
  /************************************
    自動車の推定位置から運転(移動幅と方向)を計算
    inp xEst    :推定位置
    inp icorner : 現在のコーナ
    inp xRout   :コーナの位置
    out u       :運転 u[0]:r進行幅 u[1]:θ方向幅
  ************************************/
  icorner=doControl(time,u,xEst,xRout,icorner);
  //最終コーナを過ぎると終了
  if(icorner >= MAX_CORNER) break;

  /************************************
    LandMarkの観測
    inp u     : 運転 u[0]:r進行幅 u[1]:θ方向幅
    inp xd    : 自動車の実際の位置
    out xd    : 運転u でのdt時間後の自動車の実際の位置(方向と角度にはノイズ付)
    out z     : LandMark点から自動車までの距離(ノイズ付)と観測点の位置
  ************************************/
  izn=Observation(xTrue, xd, u, XRFID, MAX_RANGE,Qsigma,Rsigma,dt,z);

  // 粒子毎の繰返し
  for(ip=0; ip <NP ; ip++) {
    for(j=0;j<DIM;j++) {
      //粒子の位置
      x[j]=px[j][ip];
    }
    //粒子の尤度
    w=pw[ip];
    /************************
     dt時間後の運転 uでの粒子の移動した位置 x
    ************************/
    factor(x,u,dt);
```

```
    //移動後の粒子の位置にQのノイズを与える
    for(j=0;j<DIM;j++) {
      // ip番目粒子の位置にノイズを追加 sqrt(Q)=[0.1 0.1 3度]
      x[j] += sqrt(Q[j][j])*pNormR(rand()/(double)RAND_MAX);
    }
    // 粒子位置と観測点との距離から尤度を求める
    for(iz=0; iz <izn /*length(z(:,1)*/; iz++) {
      for(pz=0,j=0;j<2;j++) {
        pz += pow(x[j]-z[iz][j+1],2);
      }
      /*各観測点と自動車の推定位置までの距離と
        各観測点と粒子の推定位置からの距離の誤差 dz */
      dz=sqrt(pz)-z[iz][0];
      //粒子尤度 を 誤差dzのガウス確率で積で更新
      w=w*Gauss(dz,0,sqrt(R));
    }
    //粒子の推定位置(Qのノイズ付)を格納
    for(j=0;j<DIM;j++) {
      px[j][ip]=x[j];
    }
    //尤度を格納
    pw[ip]=w;

  }

  Normalize(pw,NP);//正規化
  /************************
    /リサンプリング
  ************************/
  Resampling(px,pw,NTh,NP);

  //自動車の推定位置を全粒子の位置と尤度で更新
  for(j=0;j<DIM;j++) {
    xEst[j] = 0;
    for(ip=0;ip<NP;ip++) {
      xEst[j] += px[j][ip]*pw[ip]; //最終推定値は期待値
    }
  }
}
```

https://github.com/mabonki0725/particleNavi

まとめ

- ロボットの走行距離rと走行方向θに誤差があり、ランドマークからの距離観測dに誤差があっても、粒子フィルターで自己位置推定が出来ることが分かった。
- ランドマークからの距離観測ができないとロボットは自己位置をロストする。
- 目標とする点を指定すれば、ロボットは自動走行できることが分かった。
- 謝辞　下記のサイトのプログラムを参照した
  - http://myenigma.hatenablog.com/entry/20140628/1403956852