画像処理を用いた駐輪場混雑度判別システムの開発

06H073 服部広太郎

目次

概要

自動二輪車で通学する学生にとって駐輪場は欠かせないものである。本学では駐輪場は複数あり多数の自動二輪車を駐車できるが、それぞれが離れており、空車状況が判らないなど利便性に欠ける面が目立つ。そこで、Web 上で混雑度を確認できれば移動の無駄や無理な駐車による混雑を回避できると考え、自動二輪車用駐輪場の混雑度を Web で確認できるシステムの開発した。本システムは、設置したカメラから取得した画像をエッジ検出により処理することで台数を算出し、結果を HTML で記述し Web 上に配信するものである。実験の結果、実台数と処理結果の誤差を完全に無くす事はできなかったが、空車情報の提供という点においては、約 70% の正答率を得ることができた。今後の課題として、車両検出精度の向上、明暗の変化に対する強化、携帯電話などのモバイル機器への情報の提供を図る。

1 はじめに

自動二輪車で通学する学生にとって駐輪場は欠かせないものである。本学では駐輪場は複数あり多数の自動二輪車を駐車できるが、それぞれが離れており、空車状況が分からないなど利便性に欠ける面が目立つ。そこで、駐輪場の混雑度を判別するシステムの開発を考えた。駐輪場間での移動や通学などの際にあらかじめ駐輪場の混雑度が分かれば、移動の無駄や無理な駐車による混雑を回避できると考えたからである。また、一般的なゲートやセンサーによる台数管理ではなく、カメラで駐輪場を撮影し画像処理により台数を把握する方法を取る事により、低コストなシステムの実現を試みた。

1.1 既存のシステム

商業施設の駐車場やコインパーキングでは台数管理をするために、出入り口にゲートバーを設置したゲート方式や、各駐車スペースにフラップ板を設置したロック方式を採用している。また、一部ではカメラを利用した駐車場の混雑度判別システムを利用している例 [1] もある。

1.2 研究の目的

本研究では本学の自動二輪車用駐輪場の混雑度を Web 上で確認できるシステムの開発を目的とした。また、駐輪場が既に使用されている点、コスト面から上記のような大掛かりな設備を設置できない点から、カメラと画像処理を利用した低コストなシステムの開発も目的とした。今回の研究では本学に複数ある駐輪場のうち、私が最もよく利用する 15 号館裏の駐輪場を対象とした。

2 システムの構造

2.1 システムの概要

本研究では、駐輪場の混雑度を判定し Web 上で閲覧できるシステムを開発した。設置したカメラから画像を取得し、本システムで画像処理を行う。処理結果をログに書き込み、過去 1 時間分の結果と共に HTML として出力する。

2.2 開発環境

- ハードウェア (IBM-PC/AT 互換機)
  - CPU:Intel(R) Celeron(R) CPU 2.80GHz
  - Cache Memory:256KB
- OS：Vine Linux 4.2
- 開発言語:C 言語 / シェルスクリプト
- ネットワークカメラ (CG-WLNCMNGV2)
  - 解像度:640x480(VGA),320x240(CIF),160x120(QCIF)
  - 画像圧縮方式:JPEG

2.3 システムの動作

cron[*1]で設定した時間帯に 5 分毎に設置したカメラで撮影した画像を取得し、画像処理にて車両の有無を判定する。判定の結果をログに書き込み、HTML として出力する。車両の判定にはエッジ抽出を用いた。エッジ抽出を行い、指定した領域内のエッジの比率により駐車の有無を判定する。

画像処理による結果をログに書き込み、結果を HTML に出力する。過去 1 時間分のデータを表示させ、現在までの混雑度の推移がわかるようにした。システムの処理の流れを**図 1** に示す。

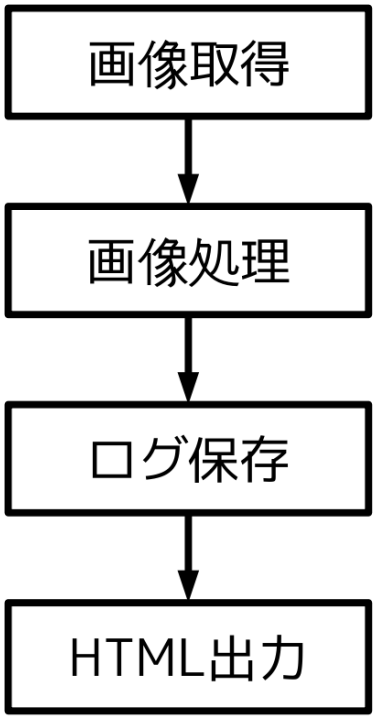

図 1　システムの流れ

[*1] cron とは Unix 系 OS において、コマンドの定時実行のスケジュール管理を行うために用いられるコマンドである。

2.4　画像処理

2.4.1　画像変換

撮影された原画像はカラー画像のため、エッジ抽出の前に濃淡画像へ変換する。濃淡画像に変換することで、路面と区別しにくい色の車両を認識しやすくする。原画像を**図 2** に示す。

図 2　原画像

次に画像を補正するため、変換された濃淡画像に対し対数変換を行う。これは影などによる明るさのムラを減らすためである。対数変換を行った場合画像の暗い部分が補正され、画像が強調される。**図 3** と**図 4** に、変換前と変換後の画像をそれぞれ示す。

図 3　濃淡画像

図 4　対数変換画像

次に変換された対数画像に対し、エッジ抽出 [2] を行う。エッジ抽出にはラプラシアン 2 フィルタ*2[3] を使用した。更に車両のエッジ部分だけを抽出するため、二値化処理を行う。車両のエッジが最も検出される 120 から 235 の間を閾値とし、条件を満たすピクセルを白 (0xff)、それ以外を黒 (0x00) とした。**図 5** に、二値化画像を示す。

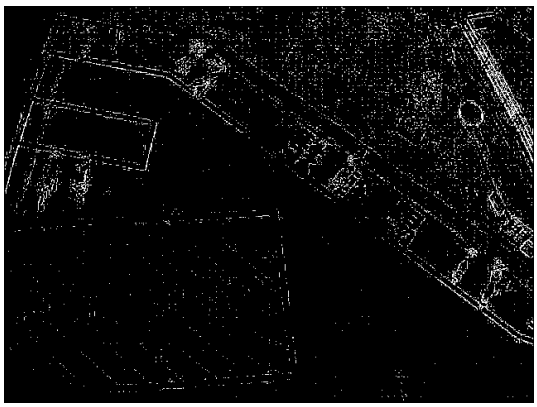

図 5　二値化画像

2.5　テンプレート画像の作成

本システムでは、画像内の駐車区域を判別するためにテンプレート画像を用いる。これは撮影した画像を編集し、判定処理の際に駐車区域のエッジだけを識別させるものである。**図 6** の黒い部分が駐車領域として指定している部分である。

図 6　テンプレート画像

2.6　初期値の設定

本システムではエッジ検出の際にノイズを完全に除去するのが不可能なため、空車時の画像を処理した際の数値を初期値として設定する。空車時の画像は、既に運用されている駐輪場から撮影するのは困難なため、フリーの画像編集ソフトを用いて作成した。

次に、車両が多く駐車されている時の画像を処理して 1 台あたりの平均ピクセル数を求めて判定処理時の基準値とする。

*2 ラプラシアン 2 フィルタとは、着目画素の近傍の画素を 2 次微分値を係数倍して着目画素から引くことにより、画像をシャープにするフィルタである。

2.7　判定処理

テンプレート画像の駐車区域内でエッジ検出されたピクセル数をカウントする。カウンタの数値から初期値を引き、1 台あたりの平均ピクセル数で割る事で台数を算出する。更に満車時の台数で割る事でパーセンテージを算出する。ここでの満車時とは厳密な台数ではなく、混雑時の台数からおおよその駐車上限を予想したものである。

2.8　ログ書込

画像処理により出た結果をログに書き込む。

2.9　HTML 出力

ログを読み込んで HTML として出力する。過去 5 分毎、1 時間分の結果を出力することで、現在までの混雑度の推移が分かるようにした。出力画面を**図 7** に示す。

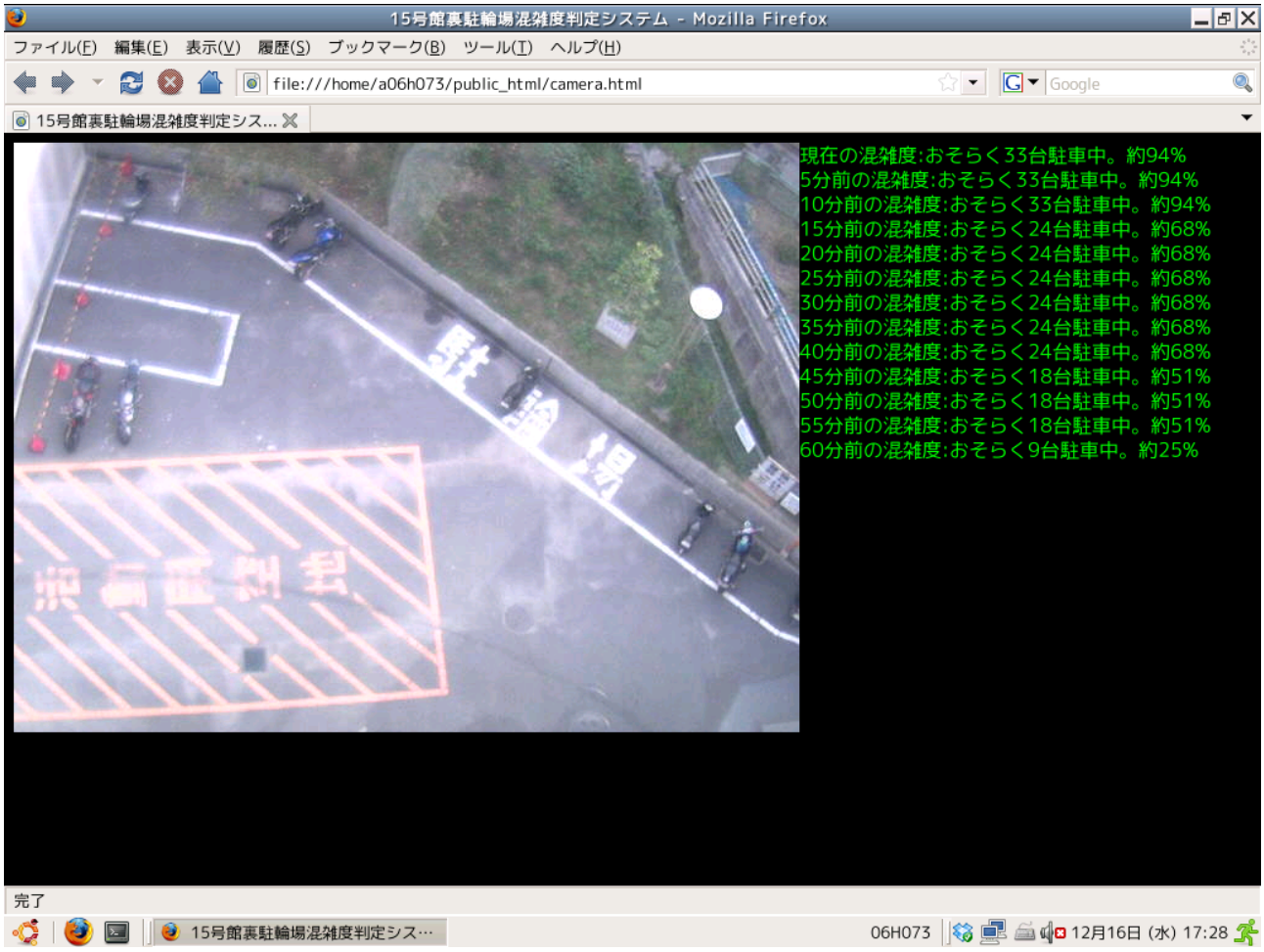

図 7　出力画面

3　結果と考察

3.1　結果

実権を行うに当たり、以下の条件を設定し、雲の出ている晴れの日に 10:00 時から 17:00 時までシステムを稼動した。

- 初期値を 3500 とする。
- 1 台当たりのピクセル数を 250 とする。
- 誤差は± 3 台までとする。
- 最大台数を 5 台とする。

10:00 時から 17:00 時までの 30 分毎の結果を**図 8** に示す。

時刻	処理結果	実台数	誤差(%)
10:00	9	8	13
10:30	10	11	10
11:00	9	17	47
11:30	20	20	0
12:00	18	21	14
12:30	25	26	4
13:00	24	26	8
13:30	33	26	27
14:00	31	26	19
14:30	36	32	13
15:00	29	31	6
15:30	29	31	6
16:00	28	30	7
16:30	21	25	16
17:00	50	23	117

図 8　結果

3.2 考察

これらの結果から、システムの利点と問題点を考察する。

3.2.1 利点

- Web 上で混雑度を確認できる
  今回の実験では、特定の時間帯において正確な混雑度の判定は出来なかったが、ユーザーにとって重要である現在駐車できるかできないかにおいてはある程度正確な情報を提供できると考える。

- コストの安さ
  今回実験で使用したのは室内用の Web カメラとカメラを固定するスタンドのみである。

- コンパクトなシステム
  カメラを固定できればどこでもシステムを実装できるため、必要な面積が小さい。

3.2.2 欠点

- 明暗の変化に対する脆弱性
  影が生じると、影とそうでない部分の境がエッジとして検出されてしまい、それが車両として認識される。結果として誤認識を生んでしまう。また本システムは駐輪場に照明装置が設置されていないため、夜間に対応していないが、学生の最も利用する時間帯ではない事から、それほど問題ではないと考える。

4　まとめ

本研究では、駐輪場の混雑度を判別するシステムを開発した。これにより駐輪場の混雑度を Web 上で確認することができるようになった。車両検出精度はまだ完全とは言えないが、空車情報をユーザーに提供する点においては、ある程度は達成できたと考える。

4.1　今後の課題

- 車両認識精度の向上
- 明暗の変化に対する強化
- 携帯電話などのモバイル機器への情報の提供

謝辞

本研究を行うにあたり、御指導及び御協力頂いた大垣 斉講師、藤井 信夫教授、情報安全工学実験室のメンバー (特に東川諒央) 及び卒業生の方々、その他見守ってくださった方々に深く感謝の意を表します。

参考文献

[1] 画像処理による照明変動等にロバストな駐車状況判定法, http://ci.nii.ac.jp/naid/110003495135
[2] 画像研究入門, http://image.onishi-lab.jp/index.html
[3] MSDN アカデミックアライアンス, http://msdn.microsoft.com/ja-jp/academic/cc998604.aspx

URI の全て 2009 年 1 月 19 日のもの

付録 A　付録 1 ソースコード

A.1　main.c

```
//画像処理プログラム
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

int log_write(char *filename, int log1, int log2){
  FILE *fp;
  fp=fopen(filename,"a");
  fprintf(fp,"%d %d\n",log1,log2);
  fclose(fp);
  return 0;
}

int ppm_read(char *filename, unsigned char *pimage){
  FILE *fp;
  if((fp=fopen(filename,"rb"))==NULL){
     printf("ファイル %s が開けません\n",filename);
     exit(-1);
  }
  fscanf(fp,"P6\n640 480\n255\n");
  fread(pimage,sizeof(char),640*480*3,fp);
  fclose(fp);
  return 0;
}

int pgm_read(char *filename, unsigned char *pimage){
  FILE *fp;
  if((fp=fopen(filename,"rb"))==NULL){
     printf("ファイル %s が開けません\n",filename);
     exit(-1);
  }
  fscanf(fp,"P5\n640 480\n255\n");
  fread(pimage,sizeof(char),640*480,fp);
  fclose(fp);
  return 0;
}

int pgm_write(char *filename, unsigned char *pimage){
```

```
  FILE *fp;
  if((fp=fopen(filename,"wb"))==NULL){
    printf("ファイル %s に書き込めません",filename);
    exit(-1);
  }
  fprintf(fp,"P5\n640 480\n255\n");
  fwrite(pimage,sizeof(char),640*480,fp);
  fclose(fp);
  return 0;
}

int main(void){

  unsigned char *image;
  unsigned char *tone;
  unsigned char *back;
  unsigned char *grey;
  unsigned char *edge1;
  unsigned char *edge2;
  int pix,per,x,y;
  int a,b,c,d;
  char *filename[64];
  FILE *fp;

  pix=0;
  c=0;
  d=0;

  image = malloc(sizeof(char)*640*480*3);
  tone  = malloc(sizeof(char)*640*480);
  back = malloc(sizeof(char)*640*480);
  grey = malloc(sizeof(char)*640*480);
  edge1 = malloc(sizeof(char)*640*480);
  edge2 = malloc(sizeof(char)*640*480);

  ppm_read("./pic/times/test1330.ppm", image);
  pgm_read("./pic/back.pgm", back);

  for(y=0;y<480;y++){
    for(x=0;x<640;x++){
      *(edge1+640*y+x)=0x00;
      *(edge2+640*y+x)=0x00;
    }
```

```
  }

  for(y=0;y<480;y++){
    for(x=0;x<640;x++){
       *(grey+640*y+x) = *(image+3*(640*y+x)+0)*0.299+
                         *(image+3*(640*y+x)+1)*0.587+
                        *(image+3*(640*y+x)+2)*0.114;
    }
  }

  for(y=0;y<480;y++){
    for(x=0;x<640;x++){
      *(tone+640*y+x) = 255*pow((double)*(grey+640*y+x)
                        /255,(double)(1/2.5));
    }
  }

  for(y=1;y<480-1;y++){
    for(x=1;x<640-1;x++){
       *(edge1+640*y+x) = (*(tone+640*(y-1)+(x-1))*-1)+
                          (*(tone+640*(y-1)+(x))*-1)+
                 (*(tone+640*(y-1)+(x+1))*-1)+
                 (*(tone+640*(y)+(x-1))*-1)+
                          (*(tone+640*(y)+(x))*8)+
                 (*(tone+640*(y)+(x+1))*-1)+
                 (*(tone+640*(y+1)+(x-1))*-1)+
                 (*(tone+640*(y+1)+(x))*-1)+
                 (*(tone+640*(y+1)+(x+1))*-1);

       if(*(edge1+640*y+x)>=120 && *(edge1+640*y+x)<=235){
         *(edge2+640*y+x) = 0xff;
       }else{
         *(edge2+640*y+x) = 0x00;
}
    }
  }

  for(x=0;x<640;x++){
    for(y=0;y<480;y++){
      if(*(back+640*y+x)==0x00 && *(edge2+640*y+x)==0xff){
    pix++;
       }
    }
```

```
  }

  printf("%d\n",pix);
  pix = (pix - 3500) / 250;
  per = (pix*100)/35;
  printf("おそらく %d 台くらい駐車中。全体の %d%%。\n",pix,per);

  pgm_write("./pic/grey.pgm",grey);
  pgm_write("./pic/tone.pgm",tone);
  pgm_write("./pic/edge1.pgm",edge1);
  pgm_write("./pic/edge2.pgm",edge2);
  log_write("./log",pix,per);

  free(image);
  free(tone);
  free(back);
  free(grey);
  free(edge1);
  free(edge2);

}
```

A.2 html.c

```
//HTML 出力プログラム
#include<stdio.h>

int main(void){

  int log[48][2];
  int a,b,c,x,y;
  int i,j;
  FILE *fp1,*fp2;
  a = 0;
  b = 1;
  c = -1;

  fp1=fopen("/home/a06h073/public_html/camera.html","w");
  fprintf(fp1,"<html>\n<head>\n<link href='./test.css'
          rel='stylesheet' type='text/css'>\n");
  fprintf(fp1,"<title>\n15 号館裏駐輪場混雑度判定システム\n
           </title>\n</head>\n<body>\n");
  fprintf(fp1,"<img src=\"test1000.jpg\" class=\"test\"
           alt=\"画像\">\n");
  fclose(fp1);

  fp2=fopen("/home/a06h073/Dropbox/sotuken/log","r");

  for(i=0;i<48;i++){
   if(fscanf(fp2,"%d %d",&x,&y) != EOF){
    log[i][0] = x; log[i][1] = y; c++;
    }
  }
  fclose(fp2);

  //printf("%d\n",c);

  fp1=fopen("/home/a06h073/public_html/camera.html","a");
  fprintf(fp1,"現在の混雑度:おそらく %d 台駐車中。約 %d%%<br>\n",
          log[c][0],log[c][1]);

  c--;

  for(i=c;i>=12;i--){
```

```
    a = b*5;
    fprintf(fp1,"分前の混雑度:おそらく %d 台駐車中。約 %d%%<br>\n",
              a,log[i][0],log[i][1]);
    b++;
 }

  fprintf(fp1,"</body>\n</html>");
  fclose(fp1);

  return 0;

}
```

A.3 picget.sh

画像取得、変換プログラム

```
#!/bin/sh

cd /home/2009/koutarou/sotuken/pic
wget -O "test.jpg" http://antcam.a4w/IMAGE.JPG?
djpeg -pnm test.jpg > test.ppm
cp test.jpg ~koutarou/public_html/
/home/2009/koutarou/sotuken/./main
/home/2009/koutarou/sotuken/./html
```