USBカメラを用いた防犯システムの構築

大西　篤史[†]　黒田　久泰[‡]

愛媛大学工学部情報工学科[†]　愛媛大学大学院理工学研究科[‡]

1. はじめに

USBカメラから取得した画像に差分処理を用いて人の検出を行う．高速化を行い，処理を行うことができるフレーム率を増やして，対象を検出できる条件の幅を向上させ，安価なカメラでも十分な検出性能を持たせることを目的とする．

2. 方法

2.1 差分法

差分法は主に背景差分法とフレーム間差分法に分けられる．また，背景差分法の改良として重み加算型背景更新法[1]がある．

2.1.1 背景差分法

背景差分法は背景画像をあらかじめ取っておき，取得したフレームと比較する方法である．対象の動きに関係なく検出できる．しかし，明るさなどで明るさの変化など背景の変化に対して弱い．

2.1.2 フレーム間差分法

フレーム間差分法は取得したフレームと前回取得したフレームを比較する方法である．背景の変化に対して強いが，動きのある対象しか検出できない．

2.1.3 重み加算型背景更新法

背景差分法に背景画像の更新を加えたもので，次のような手順で背景画像を更新する．

1. 最初に取り込んだフレーム画像を背景の初期値とする．
2. 次のフレーム画像に重み係数 k を乗算する．
3. 現在の背景画像に(1－k)を乗算する．
4. それら2つを加えた画像を新しい背景画像とする．

2.2 高速化

画像検出の処理に次のような高速化を行った．

2.2.1 ループアンローリング

ループ変数の刻み幅を大きくし，メモリアクセスのロードとストアの回数を減少させることで高速化を行う．

2.2.2 除算の除去

除算は時間がかかるので乗算とビットシフト演算に置き換える．具体的には，y = x / 3 を例にすると，x が 0≦x≦2047 を満たす整数のとき，y = x / 3 は y = x * 683 >> 11 に置き換えることができる．

2.2.3 OpenMP

OpenMP を使用してプログラムの並列化を行った．具体的には，並列化可能な各ループの前にカメラの台数と CPU 論理コア数を考慮して #pragma omp parallel for num_threads(**実行スレッド数**) を挿入した．

3. 実験

3.1 実験環境

3.1.1 マシンのスペック

実験で用いたマシンのスペックを表 1 に示す．

表1：マシンのスペック

	マシン1	マシン2
CPU	AMD Athlon64X2 BE-2350(2.1GHz)	Intel Corei7-975 (3.33GHz)
メモリ	3GB	6GB
OS	Windows 7 Professional 32bit	Windows 7 Professional 64bit

3.1.2 使用USBカメラ

実験で用いたUSBカメラのスペックを表2に示す．

表2：USBカメラのスペック

	Microsoft LifeCam Cinema	UCAM-DLC300T	UCAM-DLP130T
画素数	動画 HD 1280×720 静止画 500万画素相当	画素数 300万画素	画素数 130万画素

3.2 検出できる対象の速度

検出対象はカメラの向いている方向と垂直な方向に動くものとする．検出対象のカメラの水平広角の2等分線に垂直な方向成分v[m/sec]を求める．そのためには以下のパラメータが必要となる．

検出処理の1ループの時間：p [sec]

カメラの水平広角：θ[°]

検出対象がカメラの水平広角の 2 等分線上にいるときの距離：r [m]

Construction of the Security System Using USB Cameras

[†]Atsushi OHNISHI
Computer Science, Faculty of Engineering, Ehime University
[‡]Hisayasu KURODA
Graduate School of Science and Engineering, Ehime University

検出できる速度の下限

速度の下限は差分法によって異なる．

背景差分法，重み加算型背景更新法は

$$0 \leq v$$

を満たす．フレーム間差分法は閾値に依存する．検出できる速度の上限は次のようになる．

$$v < \frac{2r\tan\frac{\theta}{2}}{p}$$

3.3 結果

プログラムの流れを図 1 に示す．

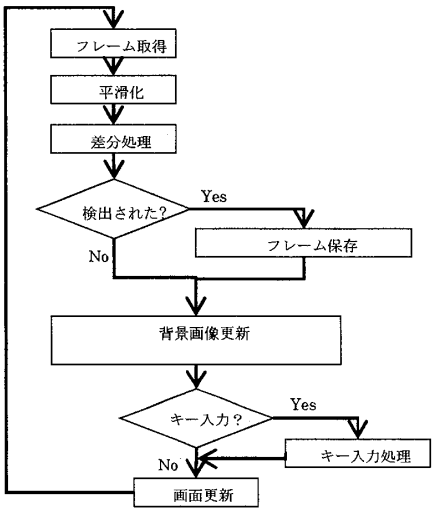

図 1：検出プログラムの流れ

表 3：マシン 1 における実行結果　（ミリ秒）

条件	フレーム取得	平滑化	差分処理	フレーム保存	背景更新	画面更新	キー入力	合計
1	1.5	59.7	30.6	-	37.9	0.2	2.1	132.1
2	1.4	41.9	29.7	-	38.5	0.1	2.1	113.7
3	1.6	28.7	21.7	-	21.1	1.0	1.8	75.9
4	1.2	26.6	23.1	35.9	19.7	0.6	1.3	108.5

表 4：マシン 2 における実行結果　（ミリ秒）

カメラ数	条件	フレーム取得	平滑化	差分処理	フレーム保存	背景更新	画面更新	キー入力	合計
1	1	0.2	13.5	4.9	-	10.1	0.0	0.0	28.7
	2	0.2	13.3	4.8	-	10.2	0.0	0.0	28.6
	3	0.2	6.1	2.2	-	4.1	0.0	0.0	12.7
	4	0.2	7.6	2.7	5.0	5.1	0.0	0.0	20.6
2	1	1.2	26.9	9.7	-	20.7	0.0	0.1	58.7
	2	1.1	26.7	9.7	-	20.6	0.0	0.1	58.3
	3	0.9	17.4	6.2	-	12.5	0.0	0.1	37.1
	4	0.5	17.5	6.1	9.8	12.2	0.0	0.1	46.3
4	1	0.9	53.9	19.4	-	40.5	0.9	1.2	116.7
	2	0.9	53.6	19.4	-	41.1	0.5	1.5	117.0
	3	0.8	24.2	9.9	-	16.2	0.1	0.2	51.5
	4	0.8	25.5	10.1	17.1	17.1	0.2	1.0	74.4

それぞれの USB カメラで処理時間を計測したが，計測結果に差はなかった．カメラの解像度はすべて 640×480 として，マシン 1，2 で計測した．

まず，マシン 1 で実験を行った．使用したカメラは Microsoft LifeCam Cinema である．重み加算型背景更新法で処理を行った(表 3 条件 1)．次に平滑化の処理部分にループアンローリングと除算置換を用いて高速化を行った(表 3 条件 2)．そして，スレッド数を 2 個に増やしてから OpenMP による並列化を平滑化と差分処理，背景更新に対して行った(表 3 条件 3)．最後に毎フレーム保存を行った(表 3 条件 4)．

次にマシン 2 を使用し，カメラの数を最大 4 台まで増やして実験を行った．使用したカメラが 1 台のときは Microsoft LifeCam Cinema，2 台のときはさらに UCAM-DLC300T を加え，4 台のときはさらに Microsoft LifeCam Cinema と UCAM-DLP130T を加えた．プログラムは C 言語で記述し，コンパイラは Microsoft Visual Studio 2008 を用いた．マシン 2 では 32bit のプログラムと 64bit のプログラムを実行できるが，どちらの計測結果も変わらなかった．

表 4 に 64bit のプログラムの実行結果を示す．マシン 2 での実験条件はマシン 1 のときと変わらないが，並列化のときのスレッド数はカメラが 1 台のときは 8 個，複数台あるときはカメラの台数個である．

マシン 1 では，検出処理 1 ループの全体の時間が 75.9msec で行われているので，動画像を最大 13 フレーム/sec で処理を行うことができる．Microsoft LifeCam Cinema の水平広角は 74° で，対象との距離を 5m とすると検出できる速度の上限は 357.4km/h となる．人が歩くところを検出するのであれば，人の歩く速度を考慮して 10km/h まで検出すれば十分である．10km/h までを検出できればよいのであれば，今までより近い距離を検出することができ，このときの検出できる距離は 0.1m までとなる．

マシン 2 のカメラ数が 1 のときは 12.7msec で処理が行われているが，実験で使用したカメラのフレーム率は最大 30 フレーム/sec なので，処理速度が 33.3msec 以下のときにはすべて同じ検出性能であった．

4. まとめ

高速化によって，検出できる対象との距離の幅，対象物の移動速度の幅，1 台のパソコンで処理できるカメラの台数を増やすことが可能となり，安価な USB カメラを用いた防犯システムでも十分な性能を得ることができた．

参考文献

[1] 藤田華那他、室内動画における状況認識、広島工業大学卒業論文(2004)