2008－CG－132 (15)
2008／8／23

ディスプレイベゼル上の付箋紙画像認識による タスク管理インタフェース

水戸 祐介 † 高井 昌彰 ††

† 北海道大学大学院情報科学研究科 †† 北海道大学情報基盤センター

近年の GTD ブームに伴い様々なタスク管理ツールが考案されている．その中でも付箋を使ったタスク管理はどこにでも貼れる手軽さから広く利用されている．しかし付箋に書いたタスクとコンピュータで管理するタスクの同期を取る必要があることや，付箋が剥がれて紛失してしまうことなどの問題があった．本研究ではコンピュータの液晶ディスプレイに貼られた付箋を画像処理によって認識しコンピュータに取り込むことで，付箋に対するディスプレイ内からの情報付加や付箋を介した計算機とのインタラクションを実現する方法を提案する．ユーザは単に付箋を貼る，剥がすなどの動作によって取り込んだ付箋データを操作することが可能になる．

To-Do List Management Interface by Recognizing Sticky Notes on the Display Bezel

Yusuke MITO † Yoshiaki TAKAI ††

† Graduate school of Information Science and Technology, Hokkaido University

†† Information Initiative Center, Hokkaido University

Abstract One of the most common methods for personal task management is by using sticky notes on the PC's display bezel. But there is a problem that he or she soon gets beyond caring it because the sticky note is always on there, it becomes like a part of the accustomed environment. In this paper, we propose a visual interaction system for the sticky notes. Our system is to give simple augmented reality to the sticky notes for effective task management. We set a small video camera on the working desk to watch around the PC's display. The system recognizes the sticky note on the bezel and shows related information such as a reminder on the PC's screen. Whenever a user puts a sticky note on the bezel, a new task associated to the note is automatically added to a task management system inside the PC.

Keywords augmented reality, man machine interaction, task management, sticky notes

1 はじめに

近年，効率的なタスク管理による個人の生産性向上が重要視されている背景から GTD(Getting Things Done) が浸透している．様々なタスク管理手法が考案されている中，付箋を使った手法は，付箋のどこにでも貼ることが出来る手軽さや手書きによる記憶の定着のしやすさなどの理由から幅広く利用されている．付箋の利用法の中でも，作業環境として身近な計算機ディスプレイのベゼル部分に，タスクを書いた付箋を貼り付けるという方法は従来からよく行われている．この方法は実行が容易である反面，ディスプレイに付箋が貼り付いている状態に慣れてしまい徐々に内容を見なくなるという状況に陥りやすい問題点がある．

そこで本研究では，ベゼルに貼り付けた付箋に対してディスプレイ内から情報を付加することで付箋に強化現実感を与え，効果的なタスク管理インタフェースを実現することを目指す．

2 関連研究

実物体の付箋とデジタルの融合に着目した研究としては次のような例がある．

2.1 Quickies

Quickies は特殊なペンを用いた文字認識と意味解析エンジンから付箋に書かれた内容を解析し，To-Do リストやカレンダーなどにタスクや予定を自動で投入することを可能にするシステムである [1)]．また付箋 1 枚ごとに RFID をつけることで貼り付けた場所の情報も参照可能となっている．しかし専用のペンや RFID を必要とするなど，汎用性やコストの面で問題が残ると考えられる．

2.2 Post-that Notes

Post-that Notes は携帯電話のカメラで付箋を撮影しブログに投稿することで予定などを管理するシステムで

ある[2]．将来的には，撮影した付箋画像をOCRによって内容解析し，カレンダーなどに自動で予定を投入できることを目的としているが，自由な書式の手書き文字に対する現状のOCRの精度を考えると実現は難しいと思われる．

これらの先行研究を踏まえた上で，本研究では市販の付箋を加工することなく使用でき，OCRなどによる文字認識は行わずに付箋のタスク管理機能を強化することを目標とする．

3 利用イメージ

本システムは次のようなシチュエーションで利用することを想定する．

- しばらく貼ったままのタスクを書いた付箋があったが，指で触れてみると2週間前に貼ったという情報が表示された．
- 明日の会議までにやることのリストを付箋に書いてディスプレイに貼る．画面内に出てきた入力ウィンドウにプレゼンテーションファイルや文書などをドラッグアンドドロップして関連付ける．会議が終わったら付箋を剥がしてタスク終了．
- いつの間にかベゼルから付箋が落ちていたが，警告ウィンドウが出てきたので気がついた．

4 付箋のタスク管理機能強化

4.1 システム概要

本システムは一般的なPCとUSBカメラから構成されており，安価なカメラを使った画像処理によってベゼル部分の付箋紙を認識する．前提条件としてカメラは図1に示すようにディスプレイ側に向けて固定するものとする．カメラの設置角度は，後述のディスプレイ位置キャリブレーションに画面の色情報を使用するため，主にディスプレイの視野角や画面の反射程度に依存し，実験ではディスプレイの中心から左右に50度程度の角度までは正常に動作した．

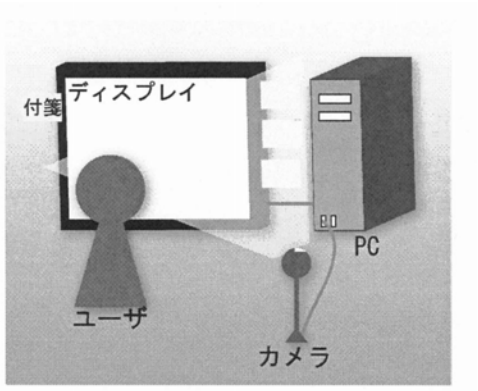


図1 システム概観

今回構築したプロトタイプシステムではPC(CPU:Core 2 Quad 2.66GHz, RAM: 4GB, OS:Windows Vista 64bit edition)，USBカメラ (Logicool Qcam Orbit AF 解像度 320x240px)，液晶ディスプレイ (17インチ，視野角:上下左右 160度) を使用した．実装言語，ライブラリとして，画像処理部分にC++/CLI，OpenCV[3]，ユーザインタフェース部分にC#とWPFを使用した．

付箋を画像認識し付加情報を表示するまでの流れは次のようになる．

- ディスプレイ位置のキャリブレーション
- 付箋画像の切り出し
- 各付箋の個体識別
- ディスプレイ上に付加情報の表示

4.2 ディスプレイ位置のキャリブレーション

付箋貼り付け位置の基準点を得るために画像内におけるディスプレイのコーナーを検出する．文献[4]を参考に画面4隅に撮影画像内で最も出現頻度が低い色のマーカを表示し (図2)，閾値処理によってそれを抽出することにより検出した (図3)．本システムではユーザに検出位置を提示し，適正な位置をユーザに判断させることで位置を確定させる．

図2 マーカ表示

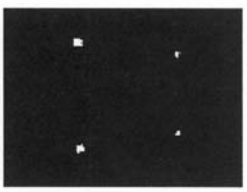


図3 検出コーナー

4.3 付箋画像の切り出し

付箋画像の切り出しは次の2段階の処理によって行った．

4.3.1 ベゼル上の付箋貼り付け部の抽出

まず4.2で得たディスプレイコーナーの座標をディスプレイのアスペクト比に応じた長方形になるようにパースペクティブ変換を行い，変換した座標からベゼル部分のみを抽出するマスク画像を作成する (図4(a))．次にマスク画像をパースペクティブ変換を施した撮影画像 (図4(b)) に被せ，ベゼル部分のみの変化を検出する．付箋が貼られる前後のマスク後画像を比較し差分をとった後，ラベリング処理[5]を行って一定サイズ以上の連続領域を抽出する (図4(c))．

4.3.2 付箋領域全体の抽出

撮影画像と付箋が貼られる前の画像にから背景差分処理を行い 2 値化した差分画像を得る．取得したベゼル上の付箋領域の位置を基準として，差分画像中の白領域をディスプレイの外側方向に探索し，連続した領域を付箋全体の領域とみなす．

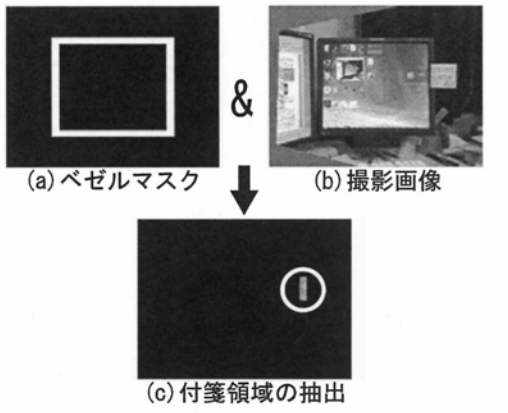


図 4　ベゼル上の付箋貼り付け部の抽出

4.4 各付箋の個体識別

4.3 までの処理でカメラから取得した 1 フレームの画像内における付箋の領域や画像情報が取得可能になる．本システムでは付箋と付箋に関連付けられた付加情報の対応付けを行うため，各フレーム間での付箋の連続性を追跡する必要がある．そのために 1 フレーム前に取得した付箋と現フレームで取得した付箋が同一であるかどうか識別するアルゴリズムを考案した．

4.4.1 個体識別処理概要

システムは 1 フレーム前に取得した付箋 (登録済付箋)，現フレームで取得した付箋 (入力付箋)，1 フレーム前に見失った付箋 (迷子付箋) の 3 つの付箋のリストを常に持つ．入力付箋と登録済付箋，入力付箋と迷子付箋の全ての付箋を後述の条件によって総当り比較を行い，それぞれのリスト内の付箋の対応付けを行う．

結果として，表 1，2 のように振り分けられる．つまり，登録済付箋と対応付けられた入力付箋は前フレームから継続して存在する付箋，迷子付箋と対応付けられた入力付箋は一時見失ったものの再検出できた付箋となる．さらにどちらとも対応付けられなかった入力付箋は新規登録付箋，入力付箋と対応付けられなかった登録済付箋は迷子付箋，入力済み付箋と対応付けられなかった迷子付箋は再度迷子付箋となる．連続して迷子付箋となった回数が一定数を超えた付箋は，ベゼルから剥がされたと判断し削除される．

表 1　入力付箋と登録済付箋の比較における振り分け

対応付け	○	×
入力	継続	新規
登録済	継続	迷子

表 2　入力付箋と迷子付箋の比較における振り分け

対応付け	○	×
入力	再検出	新規
迷子	再検出	迷子

4.4.2 付箋識別条件

個体識別は以下の条件を順に判定し，関連付ける付箋を 1 つに絞り込む．

1. 色
   付箋の色を付箋の画像領域全体の平均によって近似的に求め比較する．一定の閾値以上色が違う場合は違う付箋であるとみなす．
2. アスペクト比
   2 枚の付箋のアスペクト比を比較し，一定の閾値以上値が異なる場合は違う付箋であるとみなす．
3. 特徴点
   詳細は 4.4.3 に示す．

4.4.3 特徴点による比較

まず付箋画像に対してコーナー検出アルゴリズムを適用し特徴点を抽出する．プロトタイプシステムでは OpenCV の cvGoodFeaturesToTrack 関数を使用しコーナーを検出した．この関数は画像中から大きな固有値を持つコーナーを検出し，その中から鮮明なコーナーだけを取り出す処理を行う．

次に 2 枚の画像から検出したそれぞれの特徴点の位置を比較し，画像内での座標が近い特徴点同士を対応付ける．全ての特徴点における対応付けに成功した特徴点の割合を評価値とし，最も評価値の高い付箋のペアを同じ付箋とみなす．

4.5 付加情報の表示

認識された個々の付箋に対して付加情報をディスプレイ内から表示する．プロトタイプシステムでは，現在次の 2 つの機能が実装されている．

1. 付箋をデータに対するタグと見立てた，タスクに関連するファイルへの一時的なリンクを生成 (図 5)
   付箋を計算機内のデータに対するタグと見立てることで，付箋に対して付箋に書かれたタスクと関連するファイルを登録し，リンクによる一時的なフォルダとしての機能を持たせる．

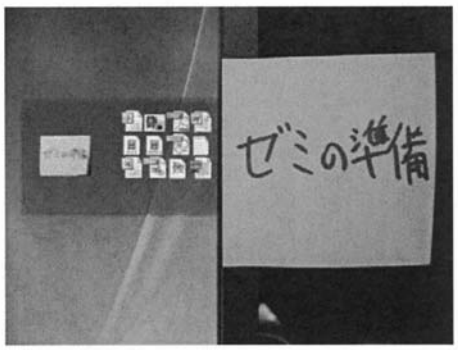


図 5　関連ファイルのリンク

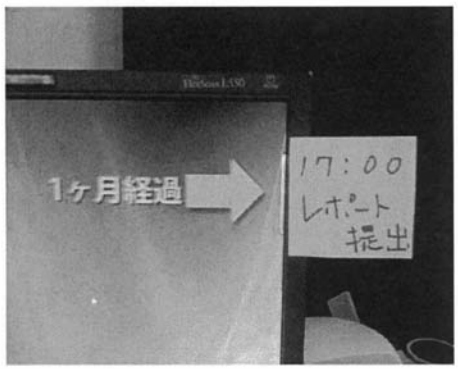


図 6　経過時間の表示

2. 付箋が貼られてからの経過時間表示 (図 6)
付箋が貼られてからの経過時間を表示することで，長時間貼ったままになっている付箋をユーザに気づかせ，タスクの消化を促す．

これらの機能によって付箋と計算機を実時間で連携させ強化現実感を与えることが可能になる．これらの付加情報はユーザが付箋を貼ることで表示され，剥がすことで削除される．ユーザの既存の付箋使用方法を妨げない形で付箋の機能を強化することで，より直感的なインタフェースを提供することができているといえる．

5　結果

今回実装したプロトタイプシステムの実行結果について処理段階ごとに述べる．実験環境のシステム構成は 4.1 節で既に示した．また実験環境におけるディスプレイとカメラの配置，ユーザとの距離は図 7 に示す．

5.1　ディスプレイ位置検出

ディスプレイ位置のキャリブレーションは，マーカーの色をカメラが正しく撮影できるかどうかが検出精度に

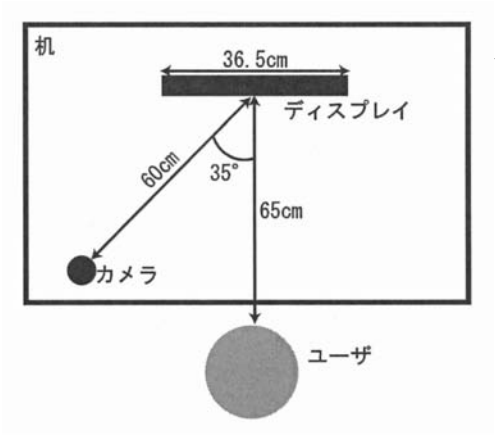
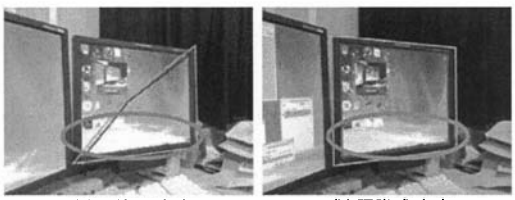


図 7　実験環境におけるディスプレイとカメラの配置

大きな影響を与えることがわかった．

蛍光灯下で適切なホワイトバランス，露出設定であればほぼ確実にディスプレイ領域を検出できたが，図 8(a) のようにカメラの自動露出機能がうまく働かない状況では，撮影画像内のマーカーの色が変化してしまうため検出できなくなる．この場合露出を手動で調整すると図 8(b) のように正しくディスプレイ領域を検出することが出来た．



(a) 認識失敗時
白が飽和している (円内)

(b) 認識成功時
露出を手動で調整

図 8　ディスプレイ領域の誤認識

また，この他に太陽光があたっている部分が撮影画像内に入る場合なども露出を正しく調整できず誤検出が起きた．

ディスプレイ検出においては最終的な判断をユーザが行うため，露出やホワイトバランスの調整によって抑えられる程度の光源の強さであれば実用上問題ないと考えられる．

5.2　付箋領域検出精度

付箋領域の検出はカメラのノイズなどの要因によって若干のゆらぎはあるものの，安定した光源下では問題なく領域を抜き出すことが出来た．

しかし付箋の背後から強い光が当たっている場合，付

箋のベゼルに重なっていない部分が光を透過してしまうためにベゼルに重なっている部分と色が異なってしまい，2 値化処理で失敗し付箋全体の領域が抽出できない場合があった．

また背景差分によって検出しているために手で付箋を移動させる際に手の部分まで付箋領域として検出してしまう問題がある．

5.3 付箋個体識別精度

付箋の個体識別精度を 2 つの方法で計測した．

静止した付箋の識別

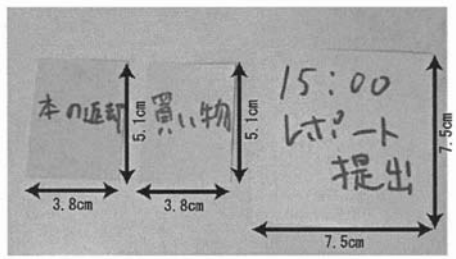


図 9　A:異なる付箋 3 枚

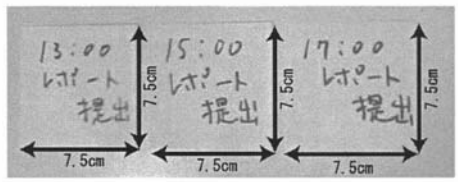


図 10　B:似た付箋 3 枚

まず図 9 と図 10 の付箋各 3 枚をベゼルに貼り付け，静止させた状態での誤認識率を調査した．図 9 は明らかに特徴が異なる付箋，図 10 は同じ付箋紙に似た文字列を書いた付箋である．

これらをそれぞれ 1 分間ベゼルに貼り付けたままにしたところ，どちらの組も一度も誤認識を起こさなかった．

結果として付箋を移動させずに静止させた状態では高精度で付箋を識別できるといえる．

移動する付箋の識別

次に図 11 と図 12 の付箋各 2 枚をベゼルに貼り付け，手で順番に付箋を移動させ貼り変えるときの誤認識率を調査した．どちらの組も同じ付箋紙を使用し，図 11 は違う文字列，図 12 は似た文字列を書いた．

ベゼル上の付箋貼り付け箇所を 3 箇所と決め，これらの付箋をローテーションさせる形で上下方向それぞれに 30 回ずつ移動させた．結果は表 3 のようになった．どちらの移動方向においても平均して 20% 以上の誤認識があり，現状では実用が難しいことがわかった．誤認識の具体的な状況として，付箋の移動後に登録済付箋であることが認識できずに新規付箋として扱われる，別の登録済付箋であると認識され付加情報が入れ替わってしまう，などの現象が発生した．

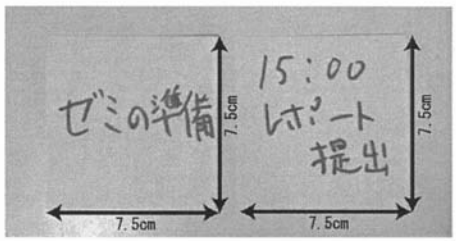


図 11　C:異なる付箋 2 枚

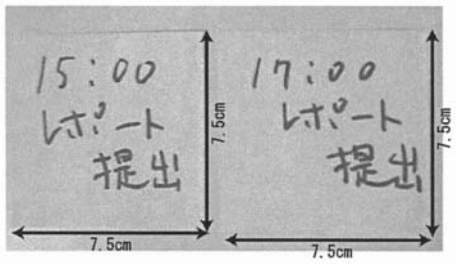


図 12　D:似た付箋 2 枚

表 3　付箋 30 回移動時の誤認識回数

	上方向	下方向	平均誤認識率
C 組	5 回	4 回	23%
D 組	5 回	4 回	23%

前者については 5.2 で述べたとおり付箋の移動時に手が付箋領域として誤検出されてしまうため，素早い動きをすると手が大きく映っている画像と手がまったくほとんど映っていない画像を比較することになり，付箋が識別できなくなったものと考えられる．付箋は常に 1 フレーム前の画像と比較するため，手や付箋の移動が緩やかな場合は関連付けを維持できていた．移動時には「ユーザの手」という大きなノイズが含まれることが避けられないため，肌色抽出などを用いて手を検出し，それに応じて付箋検出処理を一時的に中断するなどの処理を加えることで識別精度を改善する必要がある．

また後者の現象はシステム内で処理する付箋領域画像のサイズが 1 辺 25 ピクセル程度と小さいために，十分な特徴点が得られないことがあるためであると考えられる．これはより高解像度でのキャプチャや画像の先鋭化

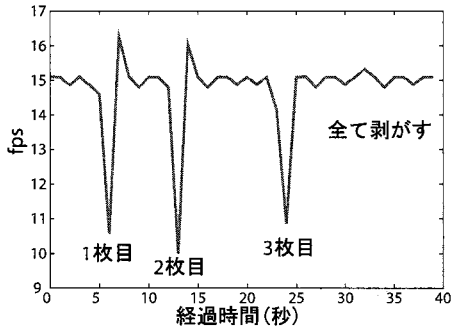


図 13　低解像度キャプチャ (320×240) でのフレームレート

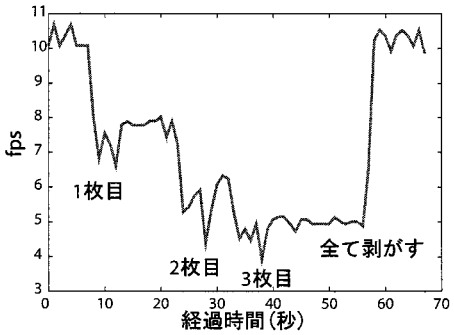


図 14　高解像度キャプチャ (960×720) でのフレームレート

などの事前処理によって，多くの特徴点を取得することで改善可能であると思われる．

5.4　処理速度

システムの主要部分である画像撮影から付加情報表示までの処理のフレームレートを計測した．本稿では全て 320×240 ピクセルの解像度で画像をキャプチャし処理を行ってきたが，ここでは 960×720 ピクセルの高解像度でキャプチャした場合の処理速度についても示す．これは高解像度化によって付箋の特徴点抽出などにおいてより多くの情報を取得できる可能性があるため，その場合のシステムに対する負荷を検証するねらいがある．

実験手順として，システムがディスプレイのキャリブレーション後にメインループに入った後，3 枚の付箋を順番に貼り付け，最後に 3 枚とも剥がすという操作を行った．まず低解像度キャプチャの結果では図 13 のように，通常時で 15fps 程度，付箋認識時に一時的に 10fps 程度まで落ちるがすぐに元のフレームレートに戻った．次に高解像度キャプチャの結果では図 14 のように，付箋が貼り付けられていない場合で 10fps 程度，付箋が 1 枚増えるごとにフレームレートが低下し 3 枚貼り付けた時点で 5fps 程度まで低下した．その後全ての付箋を剥がすと元の 10fps 程度まで処理速度が回復した．

結果として低解像度時は付箋の枚数に関わらず安定した処理速度が維持できるが，高解像度時は付箋の枚数が処理速度に大きな影響を与えることがわかった．この原因として個体識別の特徴点による比較処理が考えられる．特徴点の比較では全ての特徴点を総当りで調べるため，高解像度化によって多くの特徴点を検出した結果，マシンの処理限界を超えてしまい付箋の増加とともに処理速度が低下したと思われる．そのため現状のアルゴリズムでは 320×240 程度の解像度での処理が限界となることがわかった．

6　まとめと今後の課題

本論文では実物体である付箋に対して計算機ディスプレイ内から情報を付加することによって付箋のタスク管理機能を強化する方法について提案し，プロトタイプシステムを示した．また実験の結果，手認識による付箋の検出精度の向上や，個体識別アルゴリズムの高精度化，高速化が必要となることが明らかとなった．今後はこれらの性能向上に加えて，落下した付箋の検出や付箋に関する情報を入力しやすくするインタフェースの実現などのタスク管理機能強化，さらにタスク管理以外の用途への付箋インタフェースの応用などを考案していく予定である．

参考文献

1) Pranav Mistry and Pattie Maes, *Quickies: Intelligent Sticky Notes*, In the Proceedings of 4th International Conference on Intelligent Environments, 2008.
2) Tammy Hwang and Andres Odio, *Post-that Notes*, `http://hci.stanford.edu/cs294h/projects/post-that.doc`, 2006.
3) Intel Corporation, *OpenCV*, `http://www.intel.com/technology/computing/opencv/`
4) 吉村康弘，片山晋，古谷博史，　指先情報を用いた Vision-based Interface の提案，火の国情報シンポジウム 2005 論文集, 2005.
5) 井村誠孝，ラベリングクラス，`http://chihara.naist.jp/people/STAFF/imura/products/labeling`