アプリケーションを実世界にマッピングする動的デスクトップ

坂本拓也[†1]　伊藤栄信[†1]　二村和明[†1]

スマートフォンなどのモバイル機器により，必要に合わせていつでもどこでも情報やサービスにアクセスできるようになった．一方で，アプリケーションやサービスを使用者自身でインストールしたり，キーワードを入力して検索したりするなどの手順を踏む必要がある．そのため，居場所など使用者のコンテキストをもとにアプリケーションを提供する技術が実現されている．しかし，把握可能なコンテキストは限られているため，提供するアプリケーションを完全に絞り込むのは難しい．

そこで，カメラ入力した実世界映像にアプリケーションを重畳表示することで，アプリケーションへのアクセスを容易にするデスクトップシステムを開発した．本システムでは，その時々に得られる実世界の情報とアプリケーションの関連性を算出して対応付けることで，アクセスしたいアプリケーションを見つけやすくなると同時に，提示するアプリケーションを動的に変更することを実現する．

本論文では，本システムのアーキテクチャーを示すとともに，実装および評価結果について報告する．

Dynamic Desktop System that Maps Applications to Real World

TAKUYA SAKAMOTO[†1]　HIDENOBU ITO[†1]　KAZUAKI NIMURA[†1]

1. はじめに

日常生活を送ったり仕事をしたりする上で，必要な情報や受けたいサービスは，場所や時間などの状況によって異なる．携帯電話やスマートフォンなどのモバイル機器は，いつでもどこでも使用者自身の必要に合わせて情報やサービスにアクセスすることを可能にした．しかし，実際にはアプリケーションやサービスを使用者自身でインストールしたり，キーワードを入力して検索したりするなどの手順を踏む必要がある．

そこで，スマートフォンが具備するさまざまなセンサーで人の居場所を把握し，その場所に合ったサービスを提供するさまざまな方法が提案および実現されている．たとえば，動的デスクトップシステム[1]では，人の居場所や時刻に応じてスマートフォン上にアプリケーションをプッシュ配信し，デスクトップマネージャーにより表示するアプリケーションを切り替えることで，その人の状況に応じたアプリケーションにアクセスしやすくする．

この技術は，場所などのコンテキストに対して，情報やアプリケーションの対応付けを作成することがポイントである．まず，コンテキストを定義し，そのコンテキストに対応するアプリケーション群を作成する．そして，スマートフォンのセンサー情報の変化から認識したコンテキストをもとに対応するアプリケーション群を取得して提示することで利便性を実現する．この方法により，小さい粒度でコンテキストを定義して，そのコンテキストに対応するアプリケーション群を数個に限定すれば，使用者はアプリケーションを容易に選択できる．

しかし，実際には，GPS や Wi-Fi 測位から識別できる居場所や，スケジューラーに登録された予定などの限られた情報をもとにコンテキストを判断することになるため，コンテキストを認識できる粒度は限られる．たとえば，会議で，詳細な進行予定を作成してコンテキストを定義し，それに合わせて資料アプリケーションを対応付けたとしても，実際の進行と予定がずれたり，途中に予定外の話題が入ったりすることはよくあるが，それを認識することは難しい．そのため，コンテキストの粒度を大きくして，一つの会議全体をコンテキストとして定義して，その会議で使用する可能性のある資料アプリケーション群を対応付ける運用をすることで，必要な資料アプリケーションを提示できない状況に陥ることを防ぐことになる．その場合，対応付ける資料アプリケーションが多くなり，動的デスクトップシステムのメリットが薄れることになる．

そこで本報告では，大きな粒度でコンテキストを定義しても，使用時にアプリケーションをフィルタリングして表示することで，必要なアプリケーションへのアクセスを容易にするシステムを提案する．本システムでは，カメラ入力などで得られた実世界の情報とアプリケーションの関連性を算出することで提示するアプリケーションをフィルタリングする．また，算出結果をもとに，実世界にアプリケーションを対応付けて，AR (Augmented Reality) 技術により実世界映像にアプリケーションを合成表示する．これらにより，カメラをとおして実世界を見ることで，適切に対応付けられたアプリケーションにアクセスすることができ

†1 (株)富士通研究所
Fujitsu Laboratories Ltd.

る．

AR は，スマートフォン上のカメラから入力された景色に重ねてコンテンツを表示することで，現実を拡張する技術である．この技術は，場所・対象物に関連する付加情報を表示もしくはコメントを書き込むことができるセカイカメラ[2]などコンシューマー分野での活用が進んでいるだけでなく，業務利用での導入[3]が始まっており，人への情報提示方法のひとつとして，重要になってきている．さらに，EPSON MOVERIO[4]や Google Glass[5]と言ったシースルー型のヘッドマウントディスプレイにより，利用シーンの拡大が進むと思われる．本システムでは，これらに対しても適用可能なアーキテクチャーとなっている．

以下，2 章で本デスクトップシステムのアーキテクチャーについて，3 章でそのアーキテクチャーに沿って行った実装について述べ，4 章でその実装を評価する．5 章で関連研究について，6 章でまとめを行う．

2. アーキテクチャー

本稿での課題は，アプリケーションを見つけやすくすることと，アプリケーションなどの提示条件を容易に動的に変更可能なシステムの実現である．

本章では，まず実世界とアプリケーションを状況に合わせて対応付けるにあたり導入した概念モデルについて説明する．その後，提案するデスクトップシステムのアーキテクチャーについて述べる．なお，本システムで取り扱うアプリケーションは，汎用的に使用可能なアプリケーションとは異なり，会議資料のようなデータと，そのデータの表示や操作などをする機能をパッケージ化して，ひとつのタスクを構成するものである．以降，違いを明確にするためにタスクと記述する．

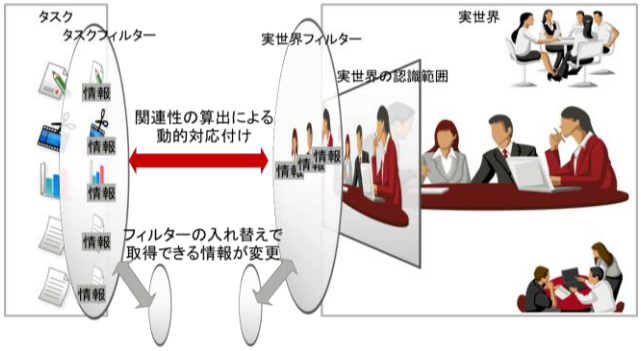

図 1　概念モデル

Figure 1 Concept Model

図 1 は提案アーキテクチャーの概念モデルを表している．同じ物を見ていても，その物への興味のあるなしなどで，その人にとっての意味が変わる．また，ある会議資料が異なる場では異なる使い方がされるように，タスクも意味が変わる．そこで，実世界フィルターとタスクフィルターの 2 つのフィルターを導入する．実世界フィルターは，実世界からカメラ等で切り出して得られる実世界の認識範囲から実世界情報を得るためのものである．また，タスクフィルターは，タスク群の一部である所有タスク群からタスク情報を得るためのものである．両フィルターともにフィルターを入れ替えることで，実世界の認識範囲や所有タスクが同じであっても，異なる情報を得ることができる．なお, 得られる情報は, 両フィルターの入れ替えに加えて，実世界の認識範囲の変更や所有タスクの入れ替えにより変わることになる．この両フィルターを通して得られる情報に合わせてタスクを実世界に対応付けることで，状況に合わせた対応付けを実現する．

次に，前述の概念モデルを実現するアーキテクチャーについて説明する．本アーキテクチャーは図 2 に示すように，AR 部，情報管理制御部，タスク実行部からなるデバイス上で動作するアプリケーションと，そのアプリケーションと連携するサーバーおよびそのサーバーに指示を出すサービス群で構成する．次に，これらの動作について説明し，本システムの特徴である情報管理制御部内の処理である動的対応付け処理の詳細を述べる．

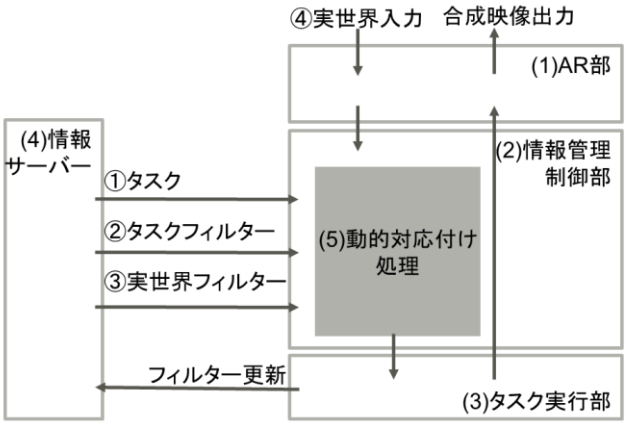

図 2　構成図

Figure 2 Structure of AR System

(1) AR 部

カメラ入力により得られた実世界映像などのセンシング情報を解析し，得られた結果を情報管理制御部に渡す．また，情報管理制御部により対応付けられたタスクを実世界映像に合成して表示する．

(2) 情報管理制御部

AR 部から渡された実世界の解析結果を情報サーバーから配信された実世界フィルターを通して実世界情報を取得する．また，情報サーバーから配信されたタスクを情報サーバーから配信されたタスクフィルターを通してタスク情報を取得する．そして，これらの得られた実世界情報およ

びタスク情報をもとに対応付け処理を行い，タスク実行基盤へのタスクの実行要求および AR 部への対応付けたタスクの表示要求を行う．

(3) タスク実行部

情報管理制御部からの要求に応じて，指定されたタスクを起動し，そのタスクを AR 部に伝える．また，タスク内に記述されたフィルター更新要求を情報サーバーに伝える．

(4) 情報サーバー

情報サーバーは，サービスとして動作しているサーバーアプリケーションからの要求やデバイス上で動作しているタスクからの要求に応じて，デバイスにタスク，実世界フィルター，アプリケーションフィルターを配信する．

(5) 動的対応付け処理

前述したように，本システムでは，情報管理制御部で実世界とタスクの対応付けを決定する．以下に示すとおり，2 種類の情報がそれぞれ 2 つの契機で変わることで対応付けを更新する．

- タスク情報
  サービス群やタスクから情報サーバー経由で追加および削除要求することで変更される所有タスク（図 2 の①）と，サービス群やタスクからの要求で情報サーバーから配信されるタスクフィルター（図 2 の②）にしたがって変わる．
- 実世界情報
  取得できる実世界情報は，カメラから入力された実世界映像などのセンシング情報（図 2 の③）と，サービス群やタスクからの要求で情報サーバーから配信される実世界フィルター（図 2 の④）にしたがって変わる．

3. 実装

本章では，前章で述べたアーキテクチャーの実装について述べる．なお，Android 搭載のデバイスを対象に実装した．

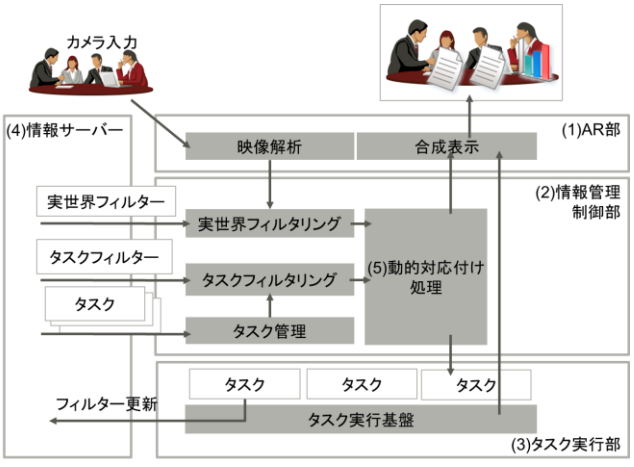

図 3　実装図

Figure 3 Implementation of AR System

(1) AR 部

AR でコンテンツを合成表示する方法として，対象物に 2 次元マーカーを貼付しておき，それをカメラから入力して識別し，実世界映像に存在する対象物に位置合わせをしてコンテンツを表示する方法がある[6][7]．本実装では，2 次元マーカーに加えて，写真から特徴点を抽出し，実世界映像とその特徴点をもとに対象物を識別できる Qualcomm®Vuforia™[8]を，映像認識に使用した．カメラ入力に応じて対象物 ID と対象物の位置座標と大きさを取得することができる．継続的にカメラ入力は行われるため，それに合わせて繰り返し映像認識を行うことになる．

実世界映像に合成表示するタスクは，HTML5 / JavaScript / CSS で記述する HTML5 アプリケーションを用いた．利用するデータと合わせて HTML5 アプリケーションを Zip 圧縮してパッケージ化した Packaged Web Apps として扱う．さらに，パッケージにはアイコンファイルも含めることで，アプリケーションアイコンの表示を可能とする．

タスクは，映像認識結果をもとに位置合わせした上で，Android 標準の HTML5 アプリケーションを表示するモジュールである WebView を使って表示する．表示内容については，実世界映像内での対象に応じて，遠距離ではアイコン，近距離では HTML5 アプリケーションの実行画面を合成して表示する．また，アイコンもしくは実行画面の一部をタッチすることで，HTML5 アプリケーションの実行画面を全画面表示できるようにした．

(2) 情報管理制御部

```
<?xml version="1.0" encoding="UTF-8"?>
<Document>
 <Target id="1">
  <keyword>whiteboard, document</keyword>
  <left>0%</:left><:top>0%</:top>
  <width>200%</width>< height>200%</height>
 </Target>
      :
</Document>
```

図 4　実世界フィルターの例

Figure 4 A example of real world filter

実世界フィルターは XML 形式で，対象物 ID，キーワードと対象物に対する表示位置と大きさの情報を記述するデータを用いる．キーワードは複数指定することが可能である．表示位置と大きさは，映像解析により得られた対象物の位置と大きさに対して相対的な数値を記述する．図 4 は実世界フィルター用データの記述例である．Target タグの id 属性で対象物 ID を定義し，値として 1 を設定している．keyword タグでキーワードを定義し，値として whiteboard

と document の 2 つを設定している．また，left/top タグで左部および上部からの表示位置を対象物の大きさに対する割合（対象物の左端を 0%とし右端を 100%とする）で定義し，ともに 0%を設定している．また，width/height タグで対象物に対する幅および高さの割合（対象物と同じ大きさを 100%とする）を定義し，ともに 200%を設定している．実世界フィルターを通すことで，AR 部で得られた対象物のキーワードと実世界映像と重ね合わせる際の表示位置と大きさを取得できる．

```
<?xml version="1.0" encoding="UTF-8"?>
<Document>
 <Task id="http://192.168.137.1/app1.zip">
  <keyword>document</keyword>
  <icon>icon.png</icon>
  <smallpage>index_small.html</smallpage>
  <toppage>index.html</toppage>
 </Task>
      :
</Document>
```

図 5　タスクフィルターの例
Figure 5 A example of task filter

タスクフィルターも XML 形式で，タスク ID，キーワードとアイコンのファイル名，表示 Web ページのファイル名を記述するデータを用いる．キーワードは複数指定することが可能である．表示用の Web ページとして，実世界映像と合成して表示するための重畳用ページと全画面表示するための全画面用ページの 2 つを記述する．図 5 はタスクフィルター用のデータの記述例である．Task タグの id 属性でタスク ID を定義し，値として，http://192.168.137.1/app1.zip を設定している．keyword タグでキーワードを定義し，値として document を設定している．icon タグでアイコンのファイル名を定義し，値として icon.png を設定している．smallpage/toppage タグで重畳用ページおよび全画面用ページのファイル名を定義し，値としてそれぞれ index_small.html と index.html を設定している．タスクフィルターを通すことで，デバイスに存在するタスクのキーワードと表示に使用するアイコンと Web ページを取得できる．

情報管理制御部は以下の処理を行う．なお，フィルターおよびタスクはサーバーからの要求に応じていつでも入れ替えることができる．

1. 対象物の取得
   映像認識の結果得られた対象物 ID のリストと対応する対象物の位置座標を取得する．この処理は，継続的なカメラからの入力に応じて，繰り返す．
2. 実世界フィルターによる実世界情報の取得
   取得した対象物 ID のリストに対応する実世界情報を取得する．1.の処理に付随して，もしくは実世界フィルターの入れ替え時に実行する．
3. タスクフィルターによるタスク情報の取得
   デバイスに存在する全タスクのタスク ID リストを取得し，それらに対応するタスク情報を取得する．所有タスクの入れ替えもしくはタスクフィルターの入れ替え時に実行する．
4. 対応付け処理
   取得した実世界情報とタスク情報の全組み合わせに対して，実世界情報に含まれるキーワードとタスク情報に含まれるキーワードの一致数を算出し，最も一致するキーワードが多い対象物とタスクを対応付ける．なお，複数の対象物に同じタスクを対応付けないように，一つの対象物に対応付けられたタスクは，他の対象物とのキーワードの一致数の算出対象から除外する．この処理は，2. 3. の処理に付随して実行する．
5. 表示位置情報を決定
   対象物の位置座標と実世界情報に含まれる表示位置と大きさの情報をもとにタスクの表示位置と大きさを決定する．また，対象物の位置座標をもとに，アイコンを表示するか，実行画面（Web ページ: smallpage）を表示するかを決定する．この処理は，1. 4. の処理に付随して実行する．
6. タスク実行部に通知
   これらの手続きにより決定されたタスクと表示位置，表示方法を，タスク実行部に伝える．この処理は，5. に付随して実行する．

(3) タスク実行部

図 6 に示す API を用意して，実世界フィルターやタスクフィルターをタスクの動作に応じて更新することを可能にした．

これらの API はタスクを構成する HTML5 アプリケーションから呼び出すことができるように JavaScript 用であり，非同期呼び出しで成功時や失敗時にコールバックされる．たとえば，getKeywordfromReal メソッドの場合，対象物 ID を targetID に，成功時に呼び出したいメソッドを success，失敗時に呼び出したいメソッドを fail に指定して呼び出すと，実行結果として，success メソッドが呼び出され，引数として，対象物 ID と対象物 ID のキーワードが渡されることになる．また，API 呼び出しを受けて，タスク実行基盤は情報サーバーにフィルターの更新要求を送信し，情報サーバーは要求を受けてフィルターを更新，各デバイスに更新されたフィルターを配信することになる．

たとえば，これらの API を使用することで，あるタスクの実行画面から操作して実世界フィルターを更新すると，

表示されている対象物から実世界フィルターをとおして得られる実世界情報のキーワードが変更されて，その対象物に重畳表示するタスクが切り替わる，といったことが実現できる．

対象物 ID を指定してキーワードを取得
getKeywordfromReal
 (targetID, success(targetID, keywords), fail(ex))
対象物 ID を指定してキーワードを設定
setKeywordtoReal
 (targetID, keywords, success(targetID), fail(ex))
タスク ID を指定してキーワードを取得
getKeywordfromTask
 (taskID, success(taskID, keywords), fail(ex))
タスク ID を指定してキーワードを設定
setKeywordtoTask
 (taskID, keywords, success(taskID), fail(ex))

図 6　キーワード変更 API
Figure 6 API for changing keywords

(4) **情報サーバー**

サーバーにタスクおよびフィルターを用意しておき，サーバー上での操作によりデバイスに対して入れ替えを要求する．また，タスク実行部からのフィルターの更新要求を受けて，更新したフィルターをデバイスに入れ替えを要求する．

(5) **動的対応付け処理**

情報管理制御部で対応付け処理として述べたとおりである．対象物もしくは実世界フィルターにしたがって変わる実世界情報と所有タスクもしくはタスクフィルターにしたがって変わるタスク情報に応じて，対応付けを更新することになる．

4. 評価

本章では，会議での利用シーンを想定して，本システム上に実際のタスクを作成してサービスを利用した評価を行う．さらに、本システムのデバイス上で動作するアプリケーションの性能を評価する．

(1) **利用評価**

以下では，利用シーンの想定，フィルターとタスクの準備に触れ，実際に利用した本システムのポイントである 4 つの動的対応付けについて述べる．また，システムの有効性や課題について考察する．

[利用シーンの想定]

利用シーンとして会議支援システムを考える．会議の以下の流れを想定した上で，これらを実現するためのタスクおよびフィルターを作成した．

1. 会議室への集合
2. アジェンダの確認
3. 議題の説明
4. 内容についての議論
5. 議事録の作成
6. 次回日時の設定
7. 解散

[フィルターとタスクの準備]

まず，1 に合わせて，アジェンダタスクなどの会議用のタスクを全て，サーバーからの要求でデバイスにインストールする．これは，業務システムのスケジューラーと連動するなどして資料を登録およびアプリケーション化した上で，場所や時間に応じて要求することを想定している．

2，3 で使用するタスクを重畳する対象物としては，人とホワイトボードとした．人を対象物としたのは，その人に関連したタスクを表示することで，わかりやすいと考えたためである．また，ホワイトボードを対象物としたのは，会議参加者がカメラを向ける方向として適切と考えたためである．カメラ入力したホワイトボードを 2 分割して，一方にはアジェンダタスク（図 7 左上）を，もう一方には議題タスク（重畳用: 図 7 中央上，全画面用: 図 7 右上）もしくは会議内容確認タスク（図 7 左下）を重畳表示する．また，議事の進行に合わせて，アジェンダタスクを使用者が操作することで，ホワイトボードに表示している議題タスクおよび会議内容確認タスクを切り替える．これは，アジェンダタスクからタスクフィルターを更新することで実現する．

図 7　タスク画面例
Figure 7 Example of tasks

また，4 を支援するために，議題タスクにメモ書き機能

を搭載した．5 として議題タスクのメモ書きをまとめて表示した上で内容を記述できる会議内容確認タスクを作成した．6 で使用する次回日時設定のためのスケジューラタスク（図 7 右下）は時計に重畳するようにした．7 に合わせて，サーバーからの要求でデバイスから資料が削除される．これは，インストール時と同様，スケジューラーや場所に応じて要求することを想定している．

ホワイトボードには 2 つのアプリケーションを重畳表示するため，マーカーとして使用する写真を 2 枚貼る．実世界フィルターのデータに含まれるキーワード（図 4 の keyword タグの記述）は，1 枚を agenda とし，もう 1 枚を document，now とした．タスクフィルターのデータに含まれるキーワード（図 5 の keyword タグの記述）は，アジェンダタスクを agenda，議題タスク（複数）と会議内容確認タスクを document，（作成者名）とした．

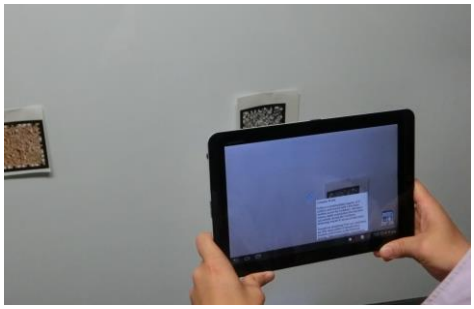

図 8　使用方法

Figure 8 Usage

[動的対応付け 1: 所有タスクの変更]

所有タスクの追加でタスク情報が変更されて，対応付けが動的に変更された例である（2 章(5)の①）．サーバーからの要求により，タスクをインストールすることで，ホワイトボードに重畳してタスクが表示されるようになる（図 9）．

図 9　所有タスクの変更

Figure 9 Changing of owned tasks

[動的対応付け 2: タスクフィルターの変更]

タスクフィルターの更新でタスク情報が変更され，その結果対応付けが変更された例である（2 章(5)の②）．アジェンダタスクの操作により，タスクフィルターのデータに含まれるキーワード（図 5 の keyword タグの記述）を変更する．具体的には，選択された議題タスクもしくは会議内容確認タスクに now キーワードを追加し，選択されなかった議題タスクもしくは会議内容確認タスクから now キーワードを削除することになる．これにより，図 10 に示すように，アジェンダタスクと議題タスクが表示されている場合，左側のアジェンダタスクの操作により，右側に表示される議題タスクが切り替わる．なお，この変更は情報サーバーを経由して，各々の会議参加者のデバイスに通知される．その結果，各々の会議参加者は使用者の一人（会議の進行役）の操作にしたがって，進行にあった会議資料に容易にアクセスすることができる．

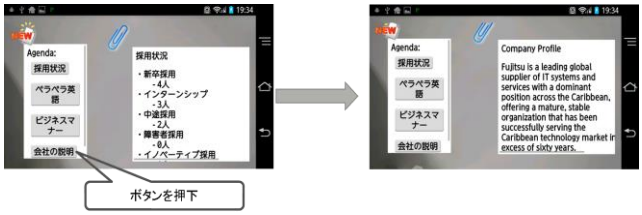

図 10　タスクフィルターの変更

Figure 10 Changing of task filters

[動的対応付け 3: センシング情報の変更]

センシング情報の更新により実世界情報が変更され，その結果対応付けが変更された例である（2 章(5)の③）．作成者にマーカーとして使用する写真を貼付して，キーワードとして，（作成者名）とした．図 11 に示すように，作成者のみが実世界映像にある状態から，作成者とホワイトボードが同時に実世界映像に含まれる状態に変えた場合，ホワイトボード上にタスクが表示されると同時に，作成者に表示されているタスクが切り替わる．これは，作成者に表示されていたタスクがホワイトボードに表示されたため，次候補となるタスクが作成者に表示されたためである．

図 11　センシング情報の変更

Figure 11 Changing of sensing information

[動的対応付け 4: 実世界フィルターの変更]

本利用シーンでは，会議内だけのシーンを対象としたため，実世界の見方を変える場面が含まれず，実世界フィルターの更新による実世界情報の変更は不要であった（2 章(5)の④）．たとえば，仕事からプライベートに切り替わるなどの対象物である人の状況が変わるようなケースの場合は，使うことが考えられる．

[考察]

図 8 で示したように，実世界映像にカメラを向けるだけで，その状況にあったタスクが表示されるためタスクの選

択が効率的になった．なお，重畳して表示するコンテンツとしてタスクを扱っているため，図 7 で示したように，議事中にメモをとったり，そのメモをまとめて会議の内容を確認に使ったり，また，スケジューラーとして使ったり，といった会議を支援するためのサービスとして提供することができた．

まず，会議の開始に合わせて会議用のタスクを情報サーバーから配信することで，その会議資料をわかりやすく表示することができた（図 9）．また，会議の進行役がアジェンダタスクを操作することで，その会議の会議者全員がその進行に合った議題タスクなどに容易にアクセスすることができた（図 10）．さらに，カメラの移動により変わった実世界映像に合わせて，表示されるタスクが切り替えられることで，簡単な操作で複数のタスクからアクセスするタスクを切り替えられた（図 11）．したがって，実世界フィルターおよびタスクフィルターにより，会議で使用する複数のタスクをフィルタリングして，使用者に提示することが実現できたと言える．また，カメラ入力した実世界映像にフィルタリングしたタスクを合成表示することで，わかりやすい提示方法となった．

一方で，以下のような課題が明らかになった．

- フィルターの作成

  本システムのポイントである動的対応付けを活用するには，適切なフィルターの作成が求められる．たとえば，評価用に作成したフィルターでは，作成者が同じである議題タスクが複数あった場合には，人に重ねて表示しようとしても，1 つしか表示されない．所有タスクがある程度絞り込めていない場合は，フィルターやフィルター更新用のタスクの作成でカバーする必要がある．

- 映像認識

  本評価では，当初の目論見とは異なり，対象物にマーカーを貼付し，そのマーカーに対してどの位置にタスクを表示するかを設定せざるを得なかった．これは，ホワイトボードや人物の写真をもとに，映像解析を行おうとしたものの，ほとんど認識できなかったためである．対象物の認識方法および準備は，本システムに限定されるものではないが，課題として残る．

- ユーザーインターフェイス

  選択は効率的になったものの，タスクを選択するごとに，スマートフォンやタブレットのカメラ入力をある方向に向けるのは手間である．デバイスとしてヘッドマウントディスプレイなどを用いた場合は，この手間は軽減される．また，ShootAR[9]のように実世界映像を静止画として取り込んで，以降はその静止画上で操作するといった工夫を組み合わせるなどで改善することも考えられる．

(2) アプリケーション性能評価

以下では，アプリケーション利用時の CPU 使用率とメモリ使用量，消費電力，表示に関する評価結果について述べる．なお，測定に使用したスマートフォンは，富士通製 ARROWS V F-04E (OS: Android 4.0.4，CPU: NVIDIA Tegra 3 AP33 1.5GHz，電池容量 2420mAh) である．

[CPU 使用率とメモリ使用量]

図 12 に実装したアプリケーションの実行時のデバイスの CPU 使用率とメモリ使用量を示す. 測定は 1 秒ごとに行なっており 20 秒ごとに同時認識させる対象物の個数を増やした．横軸の数値は同時認識した対象物の個数である．本アプリケーションでは，映像認識に使用した Qualcomm®Vuforia™の制限により同時に認識できる対象物は 5 個までとなる．結果として，CPU 使用率は同時認識対象物の個数によらず一定で，30～40%を推移している．また，メモリ使用量についても同時認識対象物の個数によらず，590M バイトで推移しており，通常のスマートフォンやタブレットで問題なく動作する使用量である．

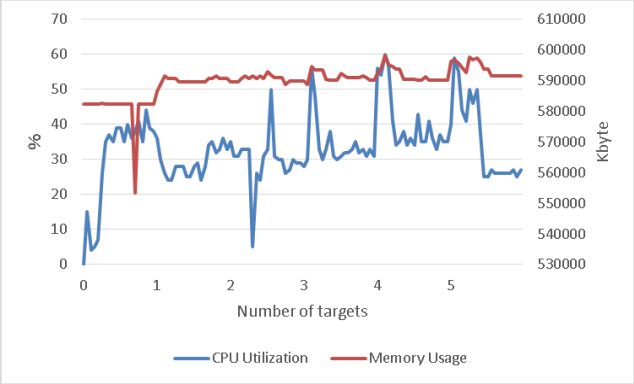

図 12　動作測定結果

Figure 12 Measurement result

[消費電力]

本アプリケーションは，ディスプレイおよび Wi-Fi をオンにした状態で継続して使用するアプリケーションであるため，消費電力は大きい．バッテリーをフル充電した状態からアプリケーションを使用しつづけた場合，2 時間 50 分でバッテリー切れとなった. 平均電流量は 14.2mA となる．

[表示]

デバイス（カメラ）を動かした際のカメラ映像とタスク表示の追従性を評価した．カメラの動きに対して実世界映像はなめらかに表示される．一方で，重畳して表示するタスクはカクついた表示となった．理由は，本アプリケーションでは，3 章で述べたように対象物の位置に追随するように WebView の移動をしているが, その処理が実世界映像のフレーム数より少ないためである．実世界映像はほぼ 60fps であるのに対して，表 1 に WebView の同時認識対象物数ごとの 1 秒あたりの表示数の測定結果として示したように，対象物の数によらず 15fps 程度となっている．違和感をなくすためには，処理を高速にするか，表示上の工夫

が必要と考えられる．

表 1　WebView フレーム数
Table 1 Frame per second of WebView

Number of targets	1	2	3	4	5
Frame per second	15.6	14.9	15.3	15.1	15.4

5. 関連研究

AR 技術を応用したデスクトップシステムとしては，実世界映像にアイコンを対応付ける手法[10]や，ヘッドマウントディスプレイでアイコンを使用してアプリケーションを起動する手法[11]などがある．これらは，使用者が手動で実世界映像にアプリケーションを対応付ける方法であり，自動で対応付ける本システムとは異なる．

ある場所に対応付けた情報を AR 技術で提示する際に，歩く，走るなどの身体動作に応じてフィルタリングするシステム[12]がある．このシステムでは，場所と情報の対応付け自体は固定であり，実世界とタスクの対応付け自体を変更する本システムとは異なる．

人の位置と時刻およびアプリケーションの利用状況に応じて，アプリケーションを推薦するシステム[13][14]についても研究がなされている．これらは，過去に利用したことがあるアプリケーションの提示手段であり，次に実行するタスクを取り扱う本システムとは異なる．

6. まとめ

カメラに写った対象物にタスクを重ねて表示することで，デバイスに存在するタスクを選び出す作業を容易にするデスクトップシステムを提案した．本システムでは，実世界フィルターおよびタスクフィルターを更新することで実世界映像および所有タスクから得られる情報を変え，その結果，実世界映像と所有タスクの対応付けを変更することができる．また，実世界映像に所有タスクを重畳して表示することで，必要なタスクを選択しやすくする．

本システムをスマートフォン上で実装し，実用的な速度と端末負荷で動作することを確認した上で，ユースケースを想定してタスクやフィルターを作成して評価を実施し，本システムの動的対応付け方式が有効であることを示した．一方で実運用に向けては，本評価で状況に合わせて手動で作成したフィルターを，自動化するなどで容易に作成できる方法が必要である．

参考文献

1)　富田達夫，角田忠信，伊藤栄信，藤野信次，飯田一朗: ヒューマンセントリックコンピューティングを実現するスマートフォンの動的デスクトップシステムの開発，情報処理学会研究報告，Vol.2012-CDS-5 No.25 (2012)
2)　Sekai Camera http://sekaicamera.com/
3)　FUJITSU Software Interstage AR Processing Server http://pr.fujitsu.com/jp/news/2013/04/24.html
4)　Epson Moverio: http://www.epson.jp/products/moverio/
5)　Google Glass: http://www.google.com/glass/start/
6)　暦本純一: 2 次元マトリックスコードを利用した拡張現実感の構成手法，日本ソフトウェア科学会 WISS’96, pp.135-145 (1996)
7)　ARToolKit http://www.hitl.washington.edu/artoolkit/
8)　Vuforia™ | Augmented Reality | Qualcomm: http://www.qualcomm.com/solutions/augmented-reality
9)　島田哲朗，樋口啓太，暦本純一: ShootAR: ユーザ姿勢を考慮したモバイル AR のための操作スタイルの提案，インタラクション 2011 論文集 (2011)
10)　坂根裕，柳沢豊，塚本昌彦，西尾章治郎: カメラ画像を利用した拡張デスクトップ環境，日本ソフトウェア科学会 SPA’98 (1998)
11)　坂根裕，塚本昌彦，西尾章治郎: HMD-TOP ヘッドマウントディスプレイを用いた拡張デスクトップシステム，情報処理学会シンポジウム 2001, pp.49-pp.50 (2001)
12)　高田大輔，小川 剛史，清川 清，竹村 治雄: 身体動作に基づき提示情報を切り替えるコンテキストアウェアなウェアラブル AR システム，ヒューマンインタフェース学会論文誌, Vol. 12, No. 1, pp. 47-pp.56 (2010)
13)　松本光弘，清原良三，沼尾正行，栗原聡: 携帯電話におけるユーザの操作パターンを用いたアプリケーション推薦方式の提案，情報処理学会 DICOMO2009 シンポジウム, pp.1149-pp.1155 (2009}.
14)　矢野幹樹，白木敦夫，梶克彦，松原茂樹，河口信夫: ユーザ生成情報を用いた携帯端末上での状況依存型サービス推薦，情報処理学会 DICOMO2010 シンポジウム, pp.221-pp.228 (2010)