TVRSJ Vol.19 No.2 pp.105-116, 2014

基礎論文

RGB-Dカメラによる環境の3次元計測に基づく実時間隠消現実感

本田 俊博[*1]　　斎藤 英雄[*1]

Real-Time Diminished Reality Based on 3D Measurement of Environment Using an RGB-D Camera

Toshihiro Honda[*1] and Hideo Saito[*1]

Abstract – **In this paper, we present a system for Diminished Reality(DR). In this system, we have two ways to perform DR: one is for a static target scene, and the other is a dynamic target scene. In the case of a static target scene, we create 3D environment model beforehand. Viewpoint Generative Learning is performed based on this model, and a database of feature descriptors is constructed. When DR is performed, feature descriptors of a smartphone image are compared with those of the database, then matched and a relative position between the smartphone and the model is computed. The model which corresponds to the position that the user touches on the smartphone is hidden and the model behind the position is drawn on the smartphone image. In the case of a dynamic target scene, feature values of a smartphone image and an RGB-D camera image are extracted for every frame, then matched and a relative position between the smartphone and the RGB-D camera is computed. Point cloud data is used instead of a model. The area specified by the user is hidden and the point cloud data is drawn on the smartphone image. We can perform real-time processing because we send the smartphone image to the PC which returns obstacle-removed images for every frame. Our experiment revealed it was possible to perform DR on the smartphone in approximately 8fps when the target scene had 3D shapes.**

Keywords : **diminished reality, real-time, smartphone, RGB-D camera, viewpoint generative learning**

1　はじめに

近年，コンピュータビジョンの分野の中で隠消現実感 (Diminished Reality) と呼ばれる技術が注目されている [1]．拡張現実感 (Augmented Reality) が実環境中にCG物体を重畳し情報を付加するのに対し，隠消現実感は実環境中の物体を視覚的に取り除くという点が異なる．隠消現実感を実行する環境には様々な条件が考えられ，どのような条件にも適切に対応できる万能な手法は存在しないと言ってもよく，隠消現実感における個々の研究は特定の条件に焦点を当てている．例えば，ユーザ視点カメラは静的か動的か，除去対象は静的か動的か，ユーザの見たいシーン，すなわち除去対象の背後（隠背景と呼ぶ）は静的か動的か，隠背景は平面か3次元形状か，実時間で動くか動かないかなどといった条件の組み合わせが考えられる．それぞれの条件を明確にした上で，隠消現実感の手法をいくつか述べる．

Zokaiらは，離れた位置から撮られた異なる3枚の静止画像を用いて，適切な背景を生成する手法を提案した [2]．この手法は，ユーザ視点カメラ，除去対象，隠背景は静的でなければならない．ただし，隠背景の3次元形状情報が必要であり，隠背景を細かく復元するほど処理時間がかかってしまう．また，障害物の境界をユーザが指定する必要がある．

Shenらは，1台のカメラで取得した画像列を用い，除去対象が移動していることを利用して隠背景を重畳する手法を提案した [3]．この手法は，ユーザ視点カメラはパン，チルト，ズームのみ可能で，除去対象は動的であることが必須である．隠背景は動的でも対応できるが，平面又は平面に近似できるものでなければならない．オフラインでの処理を想定されているため，処理時間については言及されていない．また，最初のフレームで隠背景領域をおおまかに指定しておく必要がある．

Coscoらは，事前に隠背景画像を取得しておき，除去対象の幾何モデルを基にImage Based Renderingによって隠消現実感を実現する手法を提案した [4]．この手法は，ユーザ視点カメラは動的でも対応できるが，除去対象及び隠背景は静的である必要がある．この手

[*1]慶應義塾大学
[*1]Keio University

法では，隠背景を平面又は平面に近似できる領域に分割して処理を行っているため，隠背景が複雑な3次元形状の場合は処理時間が長くなってしまう．

清水らは，距離画像カメラを用いて除去対象領域を推定し，隠背景を写す2台のカラーカメラから隠背景の画素値を取得することで隠消現実感を実現する手法を提案した [5]．この手法は，各カメラは固定されている必要があるが，除去対象及び隠背景は動的でも対応できる．ただし，ユーザが除去対象とする距離の範囲を指定しなければならず，隠背景は平面又は平面に近似できるものでなければならない．

筆者らは以前に3台のスマートフォンを用い，互いに不足した隠背景情報を補間し合うことで全てのスマートフォン上に障害物が除去された画像を表示する手法を提案した [6]．この手法は，各スマートフォン，除去対象，隠背景が動的でも対応でき，実時間で動作するが，隠背景は平面又は平面に近似できるものでなければならなかった．

別のアプローチとして，Herling らは除去対象の周辺の画素値から隠背景の画素値を推定し重畳することで除去対象を除去したように見せる手法を提案した [7]．この手法は，ユーザ視点カメラは動的でも対応できるが，除去対象及び隠背景は静的である必要がある．隠背景は平面又は平面に近似できるものでなければならないが，実時間動作が可能である．また，大雑把な除去対象領域の指定以外に必要な操作及び事前情報は不要であることも特徴である．ただし，除去対象によって隠されていない背景とは全く異なるテクスチャは生成できない．

上記に対して，本論文では，平面近似できないような3次元形状を持つ隠背景を対象に，ユーザ視点カメラ（スマートフォン）が自由に移動可能な状況で実時間で動作するような隠消現実感を実現する手法を提案する．3次元形状の把握には，カラー画像と距離画像（センサから物体までの距離が格納された画像）を同時に取得できる RGB-D カメラを用いる．本システムでは，RGB-D カメラとして，Microsoft 社の Kinect を用いる．Kinect は 2014 年現在，安価で入手しやすいデバイスであり，OpenNI[8] という Kinect を操作するライブラリも利用できる．本システムは，隠背景が静的な場合と動的な場合で，隠背景の3次元形状の取得・スマートフォンの位置姿勢推定について異なる手法を用いる．

隠背景が静的な場合，事前に Kinect を動かしながら除去対象と隠背景及びその周囲を含むように撮影し，環境の3次元モデルを作成する．このモデルを基に視点生成型学習 (Viewpoint Generative Learning)[9] を行い，特徴量のデータベースを構築する．隠消現実感を実行する際にはスマートフォン画像とデータベースの特徴量を比較し，対応付け，スマートフォンとモデルの相対位置を算出する．そしてユーザがスマートフォンをタッチした位置（除去対象とみなす）に対応するモデル中のメッシュを非表示にし，その奥のメッシュをスマートフォン画像上に重畳することで，隠消現実感を実現する．なお，ユーザから見て裏側となるメッシュは描画しないため，除去対象の反対側のメッシュは自動的に表示されない．除去対象は静的でなければならないが，除去対象が動いたかどうかの判定は行い，除去対象が物理的に取り除かれた場合は隠消現実感処理を中止する．

隠背景が動的な場合，除去対象と隠背景が映るように Kinect を設置しておく．毎フレームスマートフォン画像と Kinect 画像の特徴量を抽出し，対応付け，スマートフォンと Kinect の相対位置を算出する．モデルの代わりに Kinect で毎フレーム3次元点群を取得し，ユーザの指定した領域を非表示にして3次元点群をスマートフォン画像上に投影，重畳する．常に環境の3次元点群を取得しているので，除去対象が物理的に取り除かれた場合は自動的に隠消現実感処理が行われなくなり，入力画像をそのまま出力画像としてスマートフォン上に表示し，また再び除去対象が対象シーンに入ってきた場合には隠消現実感処理が行われる．

2 隠背景が静的の場合

2.1 システムの概要

図1にシステムの概略図を示す．本システムは1台のスマートフォンと1台のサーバにより構成される．サーバには無線 LAN ルータが接続されており，無線通信でスマートフォンとサーバ間の画像の受け渡しを行っている．また，サーバには事前に取得しておいた環境3次元モデルとその特徴量データベースが保存されている．Kinect は環境3次元モデルの取得時のみ使用し，隠消現実感処理実行時は設置する必要はない．ユーザはスマートフォンに付属しているカメラから障害物のあるユーザ視点画像を得る．そしてサーバに保存されているモデルとデータベースを用いて本手法の処理を施すことで，除去対象が除去された出力画像を得ることができる．

図2に本システムを実現するための提案手法の流れを示す．本システムはモデルとデータベースを作成するオフライン処理と，実際に隠消現実感処理を行うオンライン処理に分かれている．オフライン処理では，まず Kinect を用いて KinectFusion[10] を使用し，テクスチャ付き環境3次元モデルを作成する．その後，このモデルを多視点でレンダリングし，各視点で自然特徴を抽出，クラスタリングする視点生成型学習によっ

て特徴量データベースを作成する．

オンライン処理では，まずスマートフォン画像の自然特徴を抽出し，データベースと特徴点マッチングを行い，スマートフォン画像座標とモデル座標の 2D-3D 対応を得る．この対応を用いることで，モデル座標系をスマートフォンカメラ座標系に変換する行列 $[\mathbf{R}|\boldsymbol{t}]$ を算出できる．$[\mathbf{R}|\boldsymbol{t}]$ を用いると，スマートフォン視点でモデルを扱うことができる．

ユーザがスマートフォン上で除去対象をタッチすることにより，除去を行う領域（除去対象領域）が指定される．除去対象領域のモデルのメッシュを非表示にしてスマートフォン視点でモデルをレンダリングすることで，除去対象領域の隠背景を取得できる．この隠背景をスマートフォン画像の除去対象領域上に重畳することで隠消現実感を実現する．さらにこの時，モデルから作成した隠背景と元のスマートフォン入力画像を比較し，除去対象が物理的に除去されたと判断した場合には隠消現実感処理を中止し，入力画像をそのまま出力画像としてスマートフォン画面に表示する．なお，スマートフォン側は入力画像，入力画像のタイムスタンプ，タッチ座標，タッチ時の圧力の送信と出力画像の受信のみ行い，隠消現実感処理はサーバで行っている．

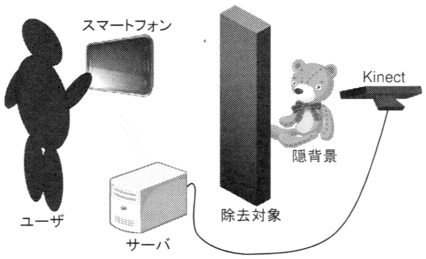


図 1　システムの概要
Fig. 1　Overview of the system

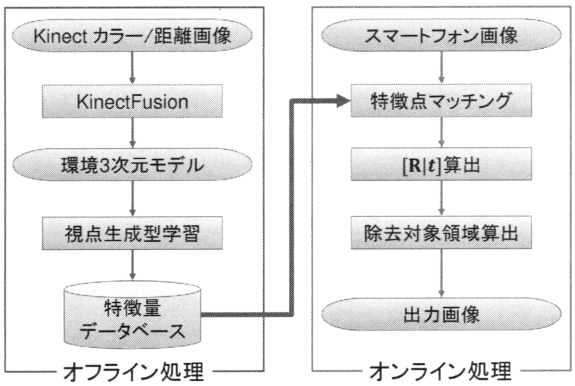


図 2　本手法の流れ（隠背景が静的の場合）
Fig. 2　Flow of the proposed method in the case of the static target scene

2.2　環境 3 次元モデル作成

本システムでは，Point Cloud Library[11] によって提供されている KinectFusion という手法を用いて環境 3 次元モデルを作成している．KinectFusion とは，Kinect で毎フレーム距離画像を取得し，Iterative Closest Point アルゴリズム [12] を用いてレジストレーションを行い，Truncated Signed Distance Function[13] を用いて 3 次元再構成を行うという手法である．3 次元再構成と並行してカラー画像の取得とその時のカメラ位置姿勢情報を保存しておくことで，テクスチャ付きの 3 次元モデルを作成することができる．

2.3　視点生成型学習

スマートフォンが自由に移動しても安定に位置姿勢推定を行うため，環境 3 次元モデルを入力とする視点生成型学習を行う．初めに，モデルを多視点でレンダリングし，GPU-SIFT[14] を用いて各視点における自然特徴点の検出及び SIFT 特徴量の抽出を行う．GPU-SIFT は，頑健な特徴抽出手法である SIFT の処理を，GPU を用いて高速化したものである．本システムではモデルの正面（モデル座標系の原点から Z 軸方向）を基準に，左右は $-90°$ から $90°$ まで $15°$ 刻みで，上下は-15° から 90°（真上）まで 15° 刻みでレンダリングしている（図 3）．ただし，真上の場合は左右 0° の 1 回のみレンダリングしているので，合計 92 視点で特徴抽出を行っている．抽出された特徴量

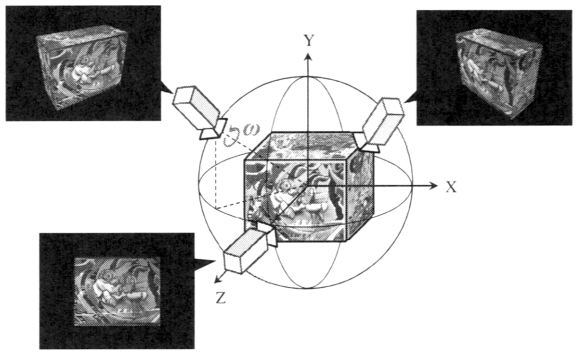


図 3　多視点でのモデルのレンダリング [9]
Fig. 3　Rendering model from many viewpoints

を全て保存するとデータベースが膨大になり，誤対応も増えると考えられるため，以下の手法を用いてデータベースを圧縮する．まず，各視点の特徴点座標をモデル座標系に逆投影していく．このとき，すでに逆投影された特徴点から一定距離内に逆投影された場合はその特徴点の repeatability スコアを 1 増やし，逆投影された特徴点の特徴量を保存する．つまり，1 つの特徴点に対し，複数の視点での特徴量を保存しておく．一定距離内に登録済みの特徴点がない場合は，新

しい特徴点として登録する．repeatability スコアが高いものを stable keypoint と呼ぶ．stable keypoint は多くの視点で観測できる特徴点ということを表す．本システムでは repeatability スコアの高い上位 2000 個の特徴点を stable keypoint として使用している．各 stable keypoint の特徴量を k-means++[15] によってクラスタリングし，特徴量データベースに保存する．本システムでは各 stable keypoint の特徴量を 8 つにクラスタリングしている．レンダリングの間隔，stable keypoint の数，クラスタリングの数は安定してカメラ位置姿勢推定が行えるように決定した．データベースには，stable keypoint のモデル座標系での 3 次元座標と，クラスタリングによって算出された 8 つの特徴量が保存されている．本システムの視点生成型学習では，1 つのモデルしか使用していないため，光沢のある物体のような，視点によって見え方が変わる環境ではうまくカメラ位置姿勢推定ができない可能性がある．

2.4 特徴点マッチング

モデルとスマートフォンの位置関係を把握するため，データベースに保存された特徴点と，スマートフォン画像から検出した特徴点とのマッチングを行う．スマートフォンは常に動いていると考えられるため，GPU-SIFT を用いて毎フレームスマートフォン画像の SIFT 特徴量を抽出する．その後，データベースに保存された特徴量と距離計算を行う．そして，最も距離が短い特徴点を対応点とする．

2.5 モデル座標系からスマートフォンカメラ座標系への変換

モデル座標系をスマートフォンカメラ座標系に変換する 3×4 行列 $[\mathbf{R}|\boldsymbol{t}]$ は，以下の式を用いて求めることができる．

$$s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{A}[\mathbf{R}|\boldsymbol{t}] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

ただし，$\mathbf{A}$ はスマートフォンカメラの内部パラメータを表す 3×3 行列，(X, Y, Z) はモデル座標，(u, v) はスマートフォン画像座標を表す．本システムでは $\mathbf{A}$ は既知とする．特徴点マッチングによって得られたスマートフォン画像座標とモデル座標の 2D-3D 対応を用いることで，$[\mathbf{R}|\boldsymbol{t}]$ を算出できる．実際には，OpenCV[16] の solvePnPRansac 関数を用いることで，$[\mathbf{R}|\boldsymbol{t}]$ の最適なパラメータを求めることができる．この関数には，誤対応が存在しても頑健に $[\mathbf{R}|\boldsymbol{t}]$ を算出できるように，RANSAC(Random Sample Consensus)[17] による外れ値除去処理が含まれている．

2.6 除去対象領域算出

どこを除去対象とみなし，隠背景を重畳するか決める必要がある．本システムでは，ユーザがスマートフォンをタッチした場所を除去対象としている．まず，何も除去していない状態のモデルを OpenGL[18] 及び 2.5 節で述べた $[\mathbf{R}|\boldsymbol{t}]$ を用いてスマートフォン視点でレンダリングする．OpenGL は，z バッファ（カメラと物体までの距離を記憶するメモリ領域）を用いてモデルを描画しており，レンダリングされた画像のある座標 (u, v) を指定すると，対応するモデルの z 座標 d を取得できる．u, v, d 及びスマートフォンカメラの内部パラメータを用いることで，(u, v) をスマートフォンカメラ座標系の 3 次元座標に変換することができる．この変換処理は，OpenGL の関数を用いている．スマートフォンカメラ座標系原点から，ユーザがタッチした画像座標に対応するスマートフォンカメラ座標系の 3 次元点へ向かう視線ベクトルを算出する．本システムで扱うモデルは三角メッシュで構成されており，交差判定法 [19] を用いて各メッシュと視線ベクトルの交差判定を行う．交差判定法は 3 次元空間において，視点からある方向に向かう視線ベクトルと三角形が交差するかどうかを判定し，交差した場合は視点から交点までの距離を算出することができる．交点が 2 つ以上あった場合に，最も手前の交点から距離 d_{th} 以内のメッシュを除去対象とみなし，非表示にする．d_{th} はユーザがスマートフォンをタッチした時の圧力によって，0 cm〜10 cm まで変化する．つまり，弱くタッチすれば細かく除去対象を指定でき，強くタッチすれば広い範囲を除去対象として指定できる．交点が 1 つの場合は，隠背景が存在しないと考えられるので，除去対象領域算出は行わない．

2.7 隠背景の重畳

除去対象領域算出後，除去対象を除去したモデルをレンダリングした画像 I_1 と，除去対象のメッシュのみからなるモデルをレンダリングした画像 I_2 を作成する．OpenGL でレンダリングされた画像の画素は，RGB チャンネルとアルファチャンネルからなる．アルファチャンネルはモデルがレンダリングされていなければ 0，レンダリングされていれば 0 以外の値が入っている．よって，I_2 のアルファチャンネルが 0 以外の画素（除去対象領域）に対応するスマートフォン入力画像の画素に，同じく対応する I_1 の画素値を上書きしていけば，除去対象領域に隠背景を重畳した出力画像を得ることができる．

2.8 除去対象が動いたかどうかの判定

本システムでは除去対象領域はモデル座標系において固定されており，実環境で除去対象が物理的に除去された場合も隠消現実感処理が行われてしまう．つま

り，すでに入力画像で隠背景が表示されていても，モデルから作成された隠背景を重畳してしまう．これを防ぐため，除去対象領域における入力画像（元画像と呼ぶ）と，モデルから作成された隠背景の画像（モデル画像と呼ぶ）との正規化相互相関 R_{zncc} を算出する．

$$R_{zncc} = \frac{\sum_{i=1}^{NUM}((I_i - \overline{I})(T_i - \overline{T}))}{\sqrt{\sum_{i=1}^{NUM}(I_i - \overline{I})^2 \times \sum_{i=1}^{NUM}(T_i - \overline{T})^2}} \quad (2)$$

ただし，I_i, T_i はそれぞれ元画像，モデル画像の除去対象領域における画素値，$\overline{I}, \overline{T}$ はそれぞれ元画像，モデル画像の除去対象領域全体の画素値の平均値，NUM は除去対象領域の画素数を表す．また $-1 < R_{zncc} < 1$ であり，1 に近づくほど，元画像とモデル画像の類似度が高いことを表す．R_{zncc} はカメラのホワイトバランスの変化のような，全体的な明るさの変動があっても値が大きく変化せず，幾何学的な位置のずれの判断に有用である．本システムでは $R_{zncc} > 0.2$ の時，除去対象が動いた（元のスマートフォン画像とモデルの隠背景が似ている）と判断し，隠消現実感処理を中止する．0.2 という値は，除去対象が動いていない時は隠消現実感処理を中止せず，除去対象が動いた場合のみ隠消現実感処理を中止するように決定した．

2.9 スマートフォンとサーバ間の画像送受信

図 4 にスマートフォンとサーバの処理の流れを示す．サーバでは，スマートフォンからの画像受信及びスマートフォンへの画像送信を行うスレッドと，隠消現実感処理を行うスレッドが存在し，それぞれ独立に処理が行われている．サーバの画像送受信スレッドでは，まず画像を受信する関数が呼ばれ，待機状態に入る．スマートフォン側で画像を送信する関数が呼ばれると，画像を受信し，入力画像を更新する．このとき，サーバの隠消現実感処理スレッドが処理中の場合，更新された入力画像を用いるのは次フレームということになる．サーバ側の画像を送信する関数が呼ばれ，かつスマートフォン側の画像を受信する関数が呼ばれていれば，出力画像を送信する．このとき，サーバの隠消現実感処理スレッドが処理中の場合，送信されるのは前フレームの出力画像ということになる．本システムではブロッキング通信を行っており，一方の受信関数が呼ばれた時，もう一方の送信関数が呼ばれるまで処理が停止する．よって，図 4 のように，サーバとスマートフォン両方とも受信の際に待機時間が発生する．

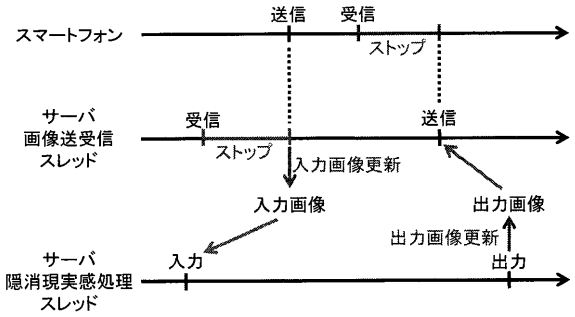


図 4 スマートフォンとサーバの処理の流れ
Fig. 4 Flow of the processing of the smartphone and the server

3 隠背景が動的の場合

3.1 システムの概要

システムの概略図は隠背景が静的の場合と同様である（図 1）．本システムは 1 台のスマートフォンと 1 台のサーバ及び 1 台の Kinect により構成される．ユーザはスマートフォンに付属しているカメラから障害物のあるユーザ視点画像を得る．Kinect は除去対象と隠背景が同時に取得できる位置に固定しておく．

図 5 に本システムを実現するための提案手法の流れを示す．まず，Kinect のカラー画像，距離画像から Kinect 視点での環境 3 次元点群を取得する．次に，Kinect カラー画像とスマートフォン画像の自然特徴を抽出し，特徴点マッチングを行う．Kinect 距離画像を用いて Kinect 側の特徴点の 3 次元座標を取得できるので，スマートフォン画像座標と Kinect カラーカメラ座標の 2D-3D 対応が求まる．この対応を用いた Kinect カラーカメラ座標系をスマートフォンカメラ座標系に変換する行列 $[\mathbf{R}|\boldsymbol{t}]$ の算出，除去対象領域の算出，隠背景の重畳は隠背景が静的の場合と同じであるので，詳細は第 2 章を参照されたい．除去対象が物理的に除去された場合は，環境 3 次元点群における除去対象も除去されるので，隠背景が静的の場合と違い，除去対象が動いたかどうかの判定は行っていない．ただし，除去対象領域は Kinect カラーカメラ座標系において固定されているので，一旦システムをリセットするまで，除去対象領域内に置かれた物体は視覚的に除去される．本システムでは，Kinect が固定されているため，Kinect カラー画像は 1 視点のものしかなく，スマートフォンが Kinect から離れると特徴点マッチング時に誤対応が増えてしまい，$[\mathbf{R}|\boldsymbol{t}]$ が正しく算出できなくなってしまう．そのため，前フレームの $[\mathbf{R}|\boldsymbol{t}]$ を用いたカメラトラッキング及び現フレームの $[\mathbf{R}|\boldsymbol{t}]$ を用いた再投影誤差評価を行い，$[\mathbf{R}|\boldsymbol{t}]$ を安定に算出している．

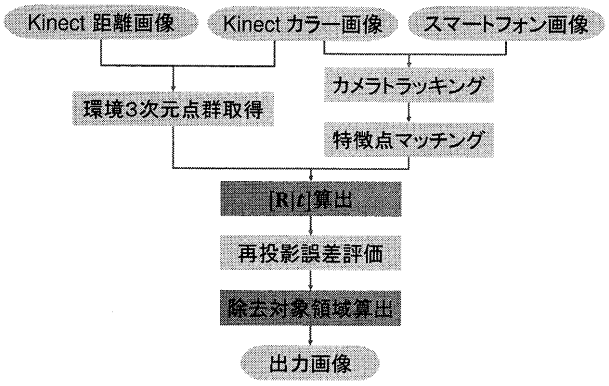


図5 本手法の流れ（隠背景が動的の場合．濃い色の処理は隠背景が静的の場合と同様）
Fig. 5 Flow of the proposed method in the case of the dynamic target scene

3.2 環境3次元点群取得

Kinectを用いると，カラー画像と距離画像を取得できる．カラー画像を撮影するカメラと，距離センサーの視点は異なるが，OpenNIを利用することにより，距離画像をカラーカメラ視点での距離画像に変換することができる．また，OpenNIには距離画像から3次元点群を算出する関数も用意されている．それぞれの点に対応するカラー画像の値を用いることで，色付きの環境3次元点群を取得できる．これは隠背景が静的の場合における環境3次元モデルに対応する．ただし隠背景が静的の場合と違い，毎フレーム環境3次元点群を取得する．距離が取得できていない画素は3次元点に変換できないので，以降の処理には用いない．

3.3 特徴点マッチング

Kinectカラー画像から検出した特徴点と，スマートフォン画像から検出した特徴点とのマッチングを行う．本システムでは，自然特徴を用いて自動的に対応付けを行う．自然特徴抽出手法にはGPU-SIFTを用いる．まず，自然特徴点を検出し，さらにその点がどのような特徴を持っているかを記述する．その後，Kinect側の特徴量とスマートフォン側の特徴量を比較し，最も似た特徴点同士を対応付ける．スマートフォン側の特徴点と対応付けたKinect側の特徴点に対応する環境3次元点群を取得することで，スマートフォン画像座標とKinectカラーカメラ座標の2D-3D対応が求まる．

3.4 カメラトラッキング

本システムではKinectカラーカメラ座標系をスマートフォンカメラ座標系に変換する行列 $[\mathbf{R}|\boldsymbol{t}]$ を安定に算出するため，スマートフォンカメラのトラッキングを行っている．トラッキングの始点となるキーフレームと，実際にトラッキングを行うキーフレーム以外の2つの場合に処理が分けられる．ここでキーフレームとは，処理を開始してから最初のフレームと，トラッキングに失敗した直後のフレームのことを指す．

キーフレームの処理について述べる．まず，Kinectカラー画像からSIFT特徴量を抽出した後，特徴点の3次元座標とその特徴量を保存しておく．これはKinectが固定されているので，キーフレーム以外ではキーフレームで取得したKinectの特徴量がそのまま使えると考えられるためである．次に，1つのKinect側の特徴点に対して，全てのスマートフォン側の特徴点と特徴量距離計算を行い，最も距離が小さい点と対応付ける．得られた対応点の組から $[\mathbf{R}|\boldsymbol{t}]$ を求める．

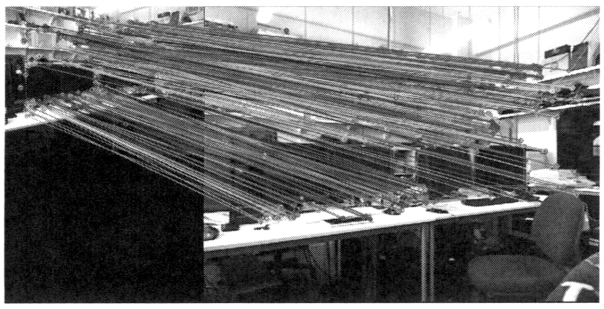
キーフレーム以外の処理について述べる．初めに，スマートフォン画像の特徴抽出を行う．処理を高速化するため，特徴点のある画素はその特徴点の番号，特徴点のない画素は -1 が入ったルックアップテーブルを作成する．次に，保存しておいたKinect側の特徴点を，前フレームの $\mathbf{A}[\mathbf{R}|\boldsymbol{t}]$ を用いてスマートフォン画像座標系に投影する．ここで，$\mathbf{A}$ はスマートフォンカメラの内部パラメータを表す．1フレームの間にスマートフォンカメラはあまり動かないと考えられるので，投影した座標の周囲にあるスマートフォン側の特徴点とのみ特徴量距離計算を行う．本システムでは，投影点を中心とした1辺61画素の正方形の範囲で特徴点と特徴量距離計算を行えば，急にスマートフォンを動かす場合を除き，$[\mathbf{R}|\boldsymbol{t}]$ が正しく算出できた．また，本システムで使用している特徴量は128次元であり，ノルムが1になるように正規化してある．そのため，特徴量距離は最大2となるが，特徴量距離が0.7以上の組の場合は $[\mathbf{R}|\boldsymbol{t}]$ の算出に適さないと考え，除外している．0.7という数値も，正しい対応を残しつつ，誤対応を減らすことができるように決定した．

あるフレームにおける，カメラトラッキングを行った場合と行っていない場合の特徴点マッチングの様子を図6に示す．左側がスマートフォン画像，右側がKinectカラー画像であり，対応点同士を線分で結んでいる．なお，RANSACを適用する前の状態のものである．図6より，カメラトラッキングを行った方が，正しいマッチングの数が多いことがわかる．

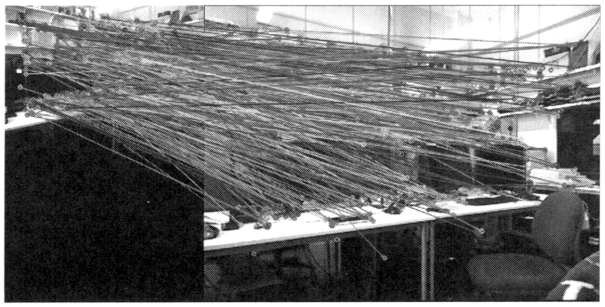
3.5 再投影誤差評価

$[\mathbf{R}|\boldsymbol{t}]$ を算出した後は，この $[\mathbf{R}|\boldsymbol{t}]$ を用いて，保存してあるKinect側の特徴点の再投影誤差評価を行う．本システムのKinect側特徴点は，5つのSIFT特徴量と，次フレームの $[\mathbf{R}|\boldsymbol{t}]$ 算出に用いるかどうかを識別する値を保持しており，再投影誤差評価によってこのパラメータを更新する．

まず，全てのKinect側の特徴点を $\mathbf{A}[\mathbf{R}|\boldsymbol{t}]$ を用いてスマートフォン画像座標系に投影する．各投影点について，最近傍に位置するスマートフォン側の特徴点を探索する．ここで最近傍というのは，特徴量距離では



(a) トラッキングあり



(b) トラッキングなし

図 6　カメラトラッキングをした場合としていない場合のマッチングの比較
Fig. 6　Comparison of matching between tracking the camera and not

なく，画像空間における特徴点間のユークリッド距離についてである．最近傍点とのユークリッド距離が再投影誤差 E [画素] となる．E がしきい値 E_{th} [画素] 以上であれば，その Kinect 側特徴点はアウトライアと判断し，次フレームの $[\mathbf{R}|\boldsymbol{t}]$ 算出に用いない．本システムでは $E_{th} = 5.0$ としている．これはなるべくインライアはインライアとして，アウトライアはアウトライアとして判定されるように決めた値である．また，4 フレーム以上連続してアウトライアと判断された特徴点は除去し，以降の $[\mathbf{R}|\boldsymbol{t}]$ 算出に用いない．再投影誤差評価の結果，$[\mathbf{R}|\boldsymbol{t}]$ の精度が高いと判断され，かつ前に Kinect 側特徴点の更新を行ってから一定フレーム経っていたら，Kinect 側特徴点を最近傍点の SIFT 特徴量を用いて FIFO（First In, First Out）で更新する．この更新によって，Kinect は固定されていながら，擬似的に複数の視点の SIFT 特徴量を保持することができ，マッチング精度の向上が期待できる．本システムでは，$E < 1.0$ の組が 50 組以上ある場合に $[\mathbf{R}|\boldsymbol{t}]$ の精度が高いと判断し，前回の更新から 10 フレーム以上経っていたら更新を行う．$[\mathbf{R}|\boldsymbol{t}]$ が正しくない時に SIFT 特徴量を更新すると誤対応が増えてしまうので，$[\mathbf{R}|\boldsymbol{t}]$ の精度評価時のアウトライア判定のしきい値は厳しく設定している．

3.6　除去対象領域算出

隠背景が静的の場合と同様に，交差判定法により除去対象領域を算出する．このとき，交差判定を行う対象であるメッシュを，環境 3 次元点群から作成する必要がある．本システムでは，Kinect 距離画像のある座標を (x, y) としたとき，

$$\{(x, y), (x+1, y), (x, y+1)\},$$
$$\{(x+1, y), (x, y+1), (x+1, y+1)\},$$
$$\{(x+1, y), (x+2, y), (x+1, y+1)\},$$
$$\ldots$$

というように規則的に 3 点を選択して三角メッシュとみなし，判定を行っている．

3.7　スマートフォンと Kinect との同期処理

本システムでは，スマートフォンのカメラ位置を正確に推定するため，スマートフォン画像と Kinect 画像との同期処理を行っている．図 7 に同期処理の概略図を示す．

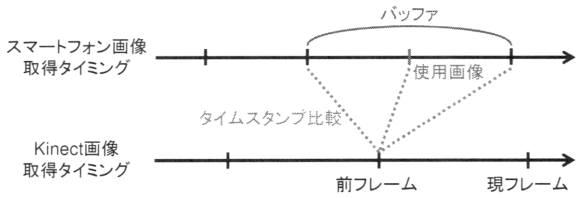


図 7　同期処理の概略図
Fig. 7　Overview of the synchronous processing

まず，スマートフォン画像と Kinect 画像それぞれにおいて，撮影された時間（タイムスタンプ）をミリ秒単位で記録する．スマートフォン画像を取得したら，バッファを更新し，タイムスタンプの新しい順に一定枚数の画像が保存されているようにする．Kinect 画像は現フレームのものを一旦保存しておき，前フレームのものを用いて隠消現実感処理を行う．前フレームの Kinect 画像のタイムスタンプに最も近いタイムスタンプを持つスマートフォン画像をバッファから選択し，隠消現実感処理に用いる．本システムでは，タイムスタンプの新しい順に 3 枚のスマートフォン画像を保存しておけば，Kinect 画像と時間的に最も近いスマートフォン画像を常に選択することができた．

以上の処理により，スマートフォン画像と Kinect 画像が別々のスレッドで取得されていても，常に時間的に最も近い組み合わせとなる画像を用いて隠消現実感処理を行うことができる．ただし，Kinect 画像は前フレームのものを用いるので，出力画像中の重畳された隠背景は，現フレームのものを用いる場合と比べ，隠消現実感処理 1 回分遅れたものとなってしまう．同様

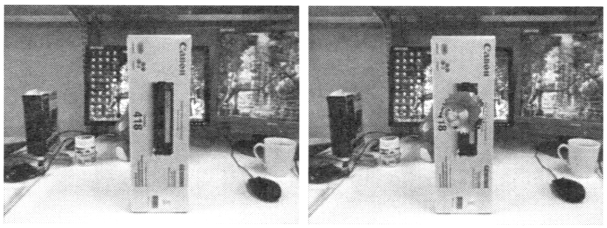
に，スマートフォン画像も最新のものが使われない可能性がある．

4 実験及び考察

本章では，5 つの項目についての実験の結果及び考察を述べる．すべての実験において，スマートフォン画像のサイズは 320×240 画素，Kinect カラー画像と距離画像のサイズは 640×480 画素である．スマートフォンカメラのホワイトバランスは「蛍光灯の下」モードである．スマートフォンのスペックは OS:Android 2.3.3，CPU:MSM8255 1GHz，RAM:512MB，実装環境:Eclipse 3.7，サーバのスペックは OS:Windows 7 64bit，CPU:Intel Core i7-2860QM 2.50GHz，RAM:16.0GB，GPU:GeForce GTX 560M，実装環境:Microsoft Visual C++ 2010 である．

4.1 実験 1 除去対象領域指定の正確さについて

隠背景が静的の場合における本システムを使用したときの入力画像と出力画像を図 8 に示す．(a)，(b) は除去対象をタッチしてすぐの画像，(c)，(d) は除去対象に沿ってスマートフォンの画面を指でなぞった画像である．なお，除去対象領域算出は隠背景が動的の場合も同じ処理であるので，隠背景が静的の場合のみ示す．



(a) 入力画像　(b) 出力画像

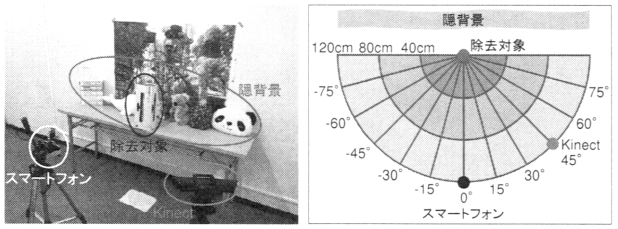
(c) 入力画像　(d) 出力画像

図 8 除去対象領域指定時の入力画像及び出力画像
Fig. 8 Input images and output images when the user touches an obstacle

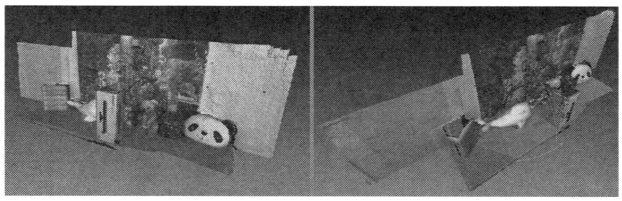
図 8 より，タッチした座標から一定の範囲が除去対象領域となり，隠背景が重畳されていることが分かる．ただし，除去対象のセグメンテーションを行っているわけではないので，除去対象の近くに除去したくないものがあると一緒に除去されてしまったり，除去対象が大きい場合は除去対象領域の指定に時間がかかってしまうといったことが考えられる．

4.2 実験 2 出力画像と実際の隠背景との類似度及び隠消現実感処理を行える範囲

隠背景が静的の場合，動的の場合それぞれにおいて，出力画像と実際の隠背景との類似度及び隠消現実感処理を行える範囲を調べる．図 9 に実験環境を示す．



(a) 実験の様子　(b) 上から見た図

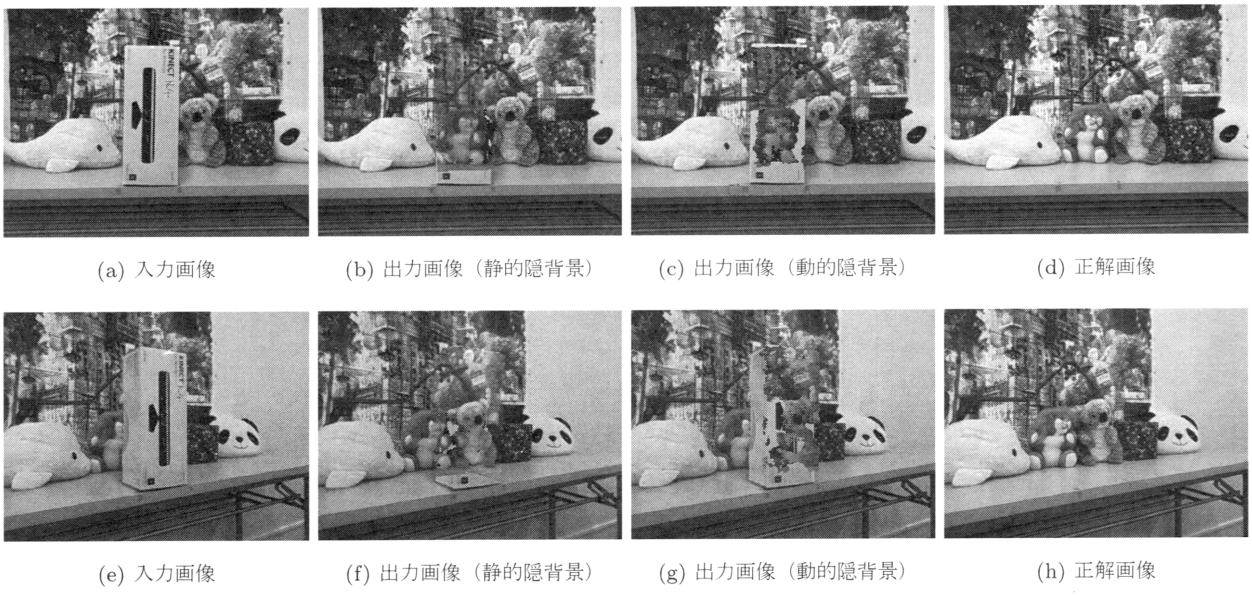


(c) KinectFusion で取得したモデル

(d) Kinect カラー画像　(e) Kinect 距離画像

図 9 実験環境
Fig. 9 Experimental environment

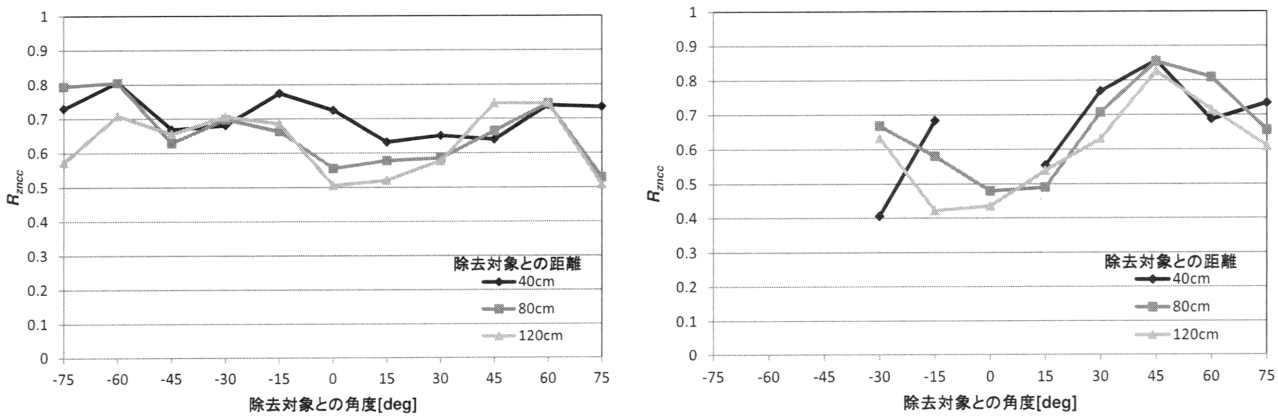
隠背景はぬいぐるみ等が置いてあり，複雑な 3 次元形状を有する机の上，除去対象は紙製の箱とした．スマートフォンは除去対象の方向を向くように固定しておき，除去対象がある場合と除去対象を取り除いた場合の 2 種類の環境を撮影する．図 9(b) のように，除去対象を中心とした半径 40 cm，80 cm，120 cm の同心円上で 15° 刻みにスマートフォンを動かしていき，除去対象が画面中央に映るように向きを固定し，隠消現実感処理を行える範囲を調べる．本実験では，各スマートフォン位置で条件を同じにするため，タッチによる除去対象領域算出は行わず，除去対象である箱を包含する直方体を除去対象領域としてあらかじめ設定しておく．類似度は，除去対象領域における，出力画像と除去対象を物理的に取り除いた状態の画像（正解画像と呼ぶ）との正規化相互相関 R_{zncc} を用いる．本システムは光学的整合性に関する処理は行っておらず，幾何学的整合性を重視しているため，全体的な明るさ



(a) 入力画像　(b) 出力画像（静的隠背景）　(c) 出力画像（動的隠背景）　(d) 正解画像

(e) 入力画像　(f) 出力画像（静的隠背景）　(g) 出力画像（動的隠背景）　(h) 正解画像

図 10　入力画像と隠背景が静的及び動的の場合の出力画像及び正解画像
Fig. 10　Input image, output image in the case of the static/dynamic target scene and the ground truth



(a) 隠背景が静的の場合の R_{zncc}　(b) 隠背景が動的の場合の R_{zncc}

図 11　隠背景が静的及び動的の場合の出力画像と正解画像との正規化相互相関 R_{zncc}
Fig. 11　Zero-mean normalized cross-correlation between the output image and the ground truth in the case of the static/dynamic target scene

の変動があっても値が大きく変わらない R_{zncc} を類似度として採用する．隠背景が動的の場合の実験では，実際には隠背景を動かさずに，隠背景が動的の場合のシステムを用いて出力画像を生成している．Kinect は図 9(b) において，120cm，45° の位置に隠背景が取得できるように設置する（図 9(d)）．図 9(e) は Kinect 距離画像であり，明るいほど距離が近いことを表す．ただし真っ白な箇所は距離が取得できていないことを表す．

図 10 に入力画像と隠背景が静的及び動的の場合の出力画像及び正解画像を示す．入力画像は図 9(b) において，上段が 0°，80cm の位置から，下段が −30°，80cm の位置から取得したものである．また，図 11 に隠背景が静的及び動的の場合の R_{zncc} を示す．ある 10 フレームの出力画像において，8 フレーム以上隠消現実感処理が成功したとみなせる場合のみ，成功したフレームにおける R_{zncc} の平均値をプロットしている．寸分違わずに隠背景を重畳することは非常に難しいため，出力画像における重畳した隠背景の各画素と，対応する実世界の隠背景との距離が 5cm 程度離れていても隠消現実感が成功したとしている．具体的な失敗例を図 12 に示す．

図 10 より，隠背景が 3 次元形状でも，スマートフォンの位置に合わせて隠背景を除去対象上に重畳できていることが分かる．隠背景が動的の場合は，静的の場合と比べ，除去対象が残っている部分がある．これは，隠背景が動的の場合は 1 視点の 3 次元点群しか取得しておらず，隠背景を取得できない領域が生じてしまう

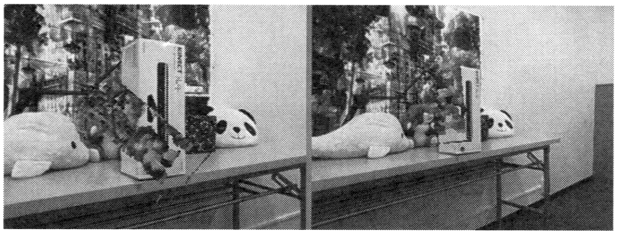


図 12 隠消現実感処理失敗例
Fig. 12 Failure examples

ためである．隠背景が静的の場合において除去対象の下部が除去できていないのも，同じく隠背景のモデルが取得できていないためである．

図 11 より，隠背景が静的の場合は全ての位置で隠消現実感処理が成功しているのに対し，隠背景が動的の場合は Kinect が置かれている位置の近くでしか処理が成功していない．これは，隠背景が静的の場合は視点生成型学習によって様々な視点での特徴を保持しているのに対し，隠背景が動的の場合では Kinect 視点の特徴しか保持していないためである．逆に多視点の特徴を保持していないがために，スマートフォンと Kinect の位置が近ければ特徴点マッチングの精度が高く，R_{zncc} が高くなると考えられる．ただし，スマートフォンと Kinect の位置が近いということは，隠背景が十分に取得できず，除去可能領域は小さくなってしまう．また，本実験において除去対象との角度が大きい場合に R_{zncc} が高くなるのは，隠背景の一部である壁面が均一な白色であり，位置がずれていても正解画像との画素値の差がほぼないためと考えられる．

4.3 実験 3 隠背景が動的の場合の本手法の有用性

実験 2 では，実際には隠背景を動かしていないため，隠背景が動的の場合に本手法が有用であるかをアンケートにより調査した．実験内容は，除去対象の裏側で手の形を変え，どのように手の形が変わったかを出力画像を見て答えてもらう，というものである．手の形は，指を全て閉じた形を「グー」，人差し指と中指を伸ばした形を「チョキ」，指を全て開いた形を「パー」と呼ぶことにする．約 1 秒経過したら手の形を変え，「グー→チョキ→パー」のように最初の手の形から 2 回手の形を変えたものを 1 セットとする．正しい順番で手の形を答えられたら正解とし，6 セット中何セット正解したかを記録する．Kinect は実験 2 と同様に，図 9(b) において，120cm，45° の位置に隠背景が取得できるように設置する．スマートフォンは 80cm の位置で 30°，15°，0° の 3 箇所に設置し，出力画像を作成した（図 13）．

被験者は 20 代の女性 2 名と男性 8 名である．調査の結果，全員が全ての位置で 6 セット全て正解することができた．図 13 を見ると，スマートフォンが Kinect に近いほど除去可能領域は小さくなっていることが分かるが，除去可能領域においてはユーザは隠背景を認識できており，本手法は有用であると言える．

(a) 0° (b) 15° (c) 30°

図 13 各スマートフォン位置における出力画像例
Fig. 13 Examples of an output image created at each smartphone position

4.4 実験 4 隠背景が動的の場合のスマートフォン画像と Kinect 画像の同期ずれ

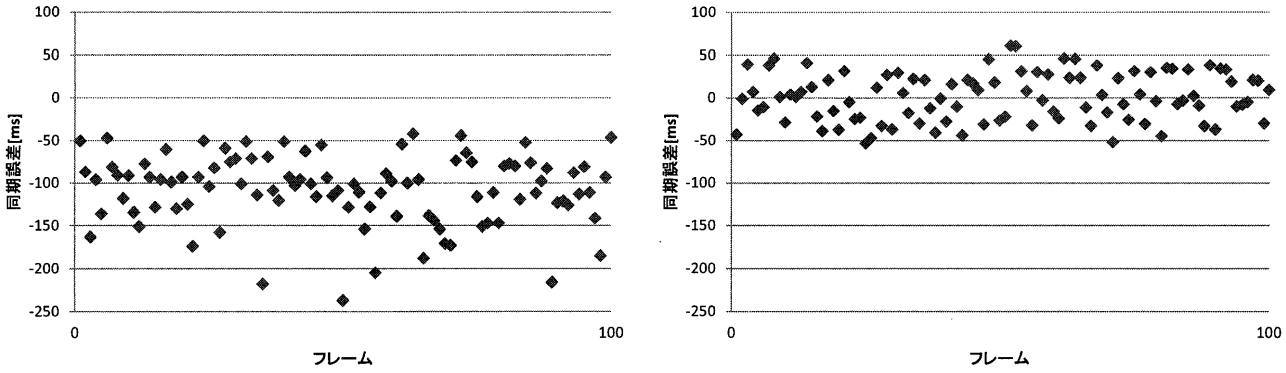
同期処理を行わない（常に最新のスマートフォン画像と Kinect 画像を入力として用いる）場合と，3.7 節に示した同期処理を行った場合のそれぞれに対し，スマートフォン画像と Kinect 画像の同期ずれを実験的に確認した．この実験では，スマートフォン画像のタイムスタンプから Kinect 画像のタイムスタンプを引いた値を同期誤差と定義する．そして，この同期誤差を 100 フレーム分計測し，プロットしたものを図 14 に示す．スマートフォン画像が Kinect 画像より後に取得されたものであれば同期誤差は正の値を取り，スマートフォン画像が Kinect 画像より前に取得されたものであれば同期誤差は負の値を取る．

図 14 より，同期処理を行わない場合は，常にスマートフォン画像は Kinect 画像よりも遅れており，同期誤差も 100 フレームの平均が −107.71 ms となっている．一方，同期処理を行った場合は，同期誤差はほぼ −50 ms から 50 ms の間の値を取っており，100 フレームの平均は 1.11 ms となった．

4.5 実験 5 主な処理にかかる時間

表 1 に隠背景が静的及び動的の場合の主な処理にかかった時間の 10 フレーム平均を示す．なお，表 1(b) の環境 3 次元点群取得及び Kinect 側特徴抽出は，カメラトラッキング時には行っていない．また，表 2 にスマートフォンとサーバのそれぞれにおいて，画像を受信する関数が呼ばれてから実際に画像データを受信し始めるまでの時間（受信待機時間），画像データを受信し始めてから受信し終わるまでにかかった時間（受信時間），画像を送信する関数が呼ばれてから実際に画像データを送信し始めるまでの時間（送信待機時間），画像データを送信し始めてから送信し終わるまでにかかった時間（送信時間）の 10 フレーム平均を示す．

サーバの隠消現実感処理スレッドの 1 サイクルは，隠背景が静的の場合は約 8fps，隠背景が動的の場合は



(a) 同期処理を行わない場合の同期誤差　　(b) 同期処理を行った場合の同期誤差

図 14　隠消現実感処理に用いられたスマートフォン画像のタイムスタンプから Kinect 画像のタイムスタンプを引いた値（同期誤差）
Fig. 14　Value of subtracting time stamp of the Kinect image from that of the smartphone image which is used in the DR processing

表 1　隠背景が静的及び動的の場合の主な処理にかかった時間 [ms]
Table 1　Main processing time in the case of the static/dynamic target scene

(a) 隠背景が静的の場合

処理内容	時間
スマートフォン側特徴抽出	23.7
特徴点マッチング	23.9
$[\mathbf{R}\|\boldsymbol{t}]$ 算出	47.1
除去対象領域算出	10.1
隠背景の重畳	10.2
除去対象が動いたかの判定	<1

(b) 隠背景が動的の場合

処理内容	時間
環境 3 次元点群取得	5.7
Kinect 側特徴抽出	72.7
スマートフォン側特徴抽出	24.2
特徴点マッチング	36.4
$[\mathbf{R}\|\boldsymbol{t}]$ 算出	48.2
再投影誤差評価	<1
除去対象領域算出	12.9
隠背景の重畳	18.1

表 2　スマートフォンとサーバそれぞれにおける画像送受信に関する処理時間 [ms]
Table 2　Processing time of sending and receiving an image for the smartphone and the server

	スマートフォン	サーバ
受信待機時間	22.1	85.3
受信時間	14.2	13.8
送信待機時間	<1	<1
送信時間	<1	<1

約 7fps であった．隠背景が静的の場合，動的の場合共に，画像送受信スレッドの 1 サイクルは約 10fps，スマートフォン画面の更新は約 10fps であった．また，視点生成型学習にかかった時間は約 20 秒であった．スマートフォンカメラで取得した画像が隠消現実感処理を経てスマートフォン画面に表示されるまでの時間（レイテンシ）は，隠背景が静的の場合，サーバがスマートフォン画像を受信する時間 13.8ms，隠消現実感処理を行う時間 115ms，スマートフォンが出力画像を受信する時間 14.2ms，スマートフォン画面に画像を表示する時間 21ms の合計約 164ms となる．ただし，実際には画像送受信スレッドと隠消現実感処理スレッドは別々に動いており，入力画像更新直後にその画像を使った隠消現実感処理が開始する，あるいは出力画像更新直後にその画像をスマートフォンに送信するとは限らず，レイテンシはより大きくなると考えられる．また，隠背景が動的の場合は，3.7 節で述べたように，最新のスマートフォン画像が使用されるとは限らず，さらにレイテンシが大きくなる可能性がある．

5　おわりに

本論文では，隠背景が 3 次元形状であっても，RGB-D カメラによって環境を計測することで，スマートフォン上で除去対象を除去し隠背景を重畳する隠消現実感を実現する手法を提案した．隠背景が静的の場合は事前に取得した環境 3 次元モデルを用いて安定に隠消現実感を実現し，隠背景が動的の場合は毎フレーム環境 3 次元点群を取得することで隠消現実感を実現することができた．主な処理をサーバで行うことで，約 8fps で処理が可能となり，実時間性を持たせることができた．

今後の課題としては，隠背景が動的の場合に，より安定かつより鮮明な隠背景の重畳を行うということが挙げられる．本システムでは，1 台の Kinect を固定した状態で使用しているため，一定の範囲しか 3 次元形

状を計測できない．また，3次元点群をそのまま画像上に投影しているため，出力があまり鮮明ではない．そこで，2台以上のRGB-Dカメラを用いて高速に鮮明な環境3次元モデルを作成することができれば，隠背景が動的でも，鮮明に隠背景を重畳することが可能であると考えられる．また，本システムでは除去対象領域はカメラ座標系において固定されているため，除去対象を動かしてしまうと除去されなくなるが，セグメンテーションの技術を用いることで，除去対象を動かしても除去対象領域を追従させ除去し続けることができると考えられる．さらにセグメンテーションを行うことで，除去対象領域算出時に一定の範囲ではなく，除去対象のみを除去対象領域として設定することも可能になると考えられる．

謝辞

本研究の一部は，科学研究費基盤研究 (S)24220004の補助により行われた．

参考文献

[1] 森 尚平，一刈 良介，柴田 史久，木村 朝子，田村 秀行，“隠消現実感の技術的枠組と諸問題～現実世界に実在する物体を視覚的に隠蔽・消去・透視する技術について～”，日本バーチャルリアリティ学会論文誌，vol.16，no.2，pp.239–250，2011.

[2] S. Zokai, J. Esteve, Y. Genc and N. Navab, “Multiview paraperspective projection model for diminished reality”, In Proceedings of the 2nd International Symposium on Mixed and Augmented Reality, pp.217–226, 2003.

[3] Y. Shen, F. Lu, X. Cao and H. Foroosh, “”Video Completion for Perspective Camera Under Constrained Motion,” In Proceedings of the 18th International Conference on Pattern Recognition, vol.3, pp.63–66, 2006.

[4] F. I. Cosco, C. Garre, F. Bruno, M. Muzzupappa and M. A. Otaduy, “Augmented touch without visual obtrusion,” In Proceedings of the 8th International Symposium on Mixed and Augmented Reality, pp.99–102, 2009.

[5] 清水 直樹，橋本 昴宗，植松 裕子，斎藤 英雄，“デプスカメラを用いたリアルタイム領域抽出による隠消現実感映像生成”，映像情報メディア学会誌，vol.66，no.12，pp.549–552，2012.

[6] 本田 俊博，斎藤 英雄，“複数のスマートフォンカメラの協調利用による実時間隠消現実感”，日本バーチャルリアリティ学会論文誌，vol.17，no.3，pp.181–190，2012.

[7] J. Herling and W. Broll, “PixMix: A real-time approach to high-quality Diminished Reality,” In Proceedings of the 11th International Symposium on Mixed and Augmented Reality, pp.141–150, 2012.

[8] Open Natural Interaction Library, http://www.openni.org/

[9] D. Thachasongtham, T. Yoshida, F. de Sorbier and H. Saito, “3D object pose estimation using viewpoint generative learning,” In Proceedings of the 18th Scandinavian Conference on Image Analysis, pp.512–521, 2013.

[10] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, “KinectFusion: Real-time dense surface mapping and tracking,” In Proceedings of the 10th International Symposium on Mixed and Augmented Reality, pp.127–136, 2011.

[11] R. B. Rusu and S. Cousins, “3D is here: Point cloud library (PCL),” In International Conference on Robotics and Automation, pp.1–4, 2011.

[12] P. Besl and N. McKay, “A method for registration of 3D shapes,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.14, no.2, pp.239–256, 1992.

[13] B. Curless and M. Levoy, “A volumetric method for building complex models from range images,” In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, pp.303–312, 1996.

[14] S. N. Sinha, J. M. Frahm, M. Pollefeys and Y. Genc, “GPU-based video feature tracking and matching,” In Workshop on Edge Computing Using New Commodity Architectures, vol.278, pp.4321, 2006.

[15] D. Arthur and S. Vassilvitskii, “k-means++: The advantages of careful seeding,” In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.1027–1035, 2007.

[16] Open Computer Vision Library, http://sourceforge.net/projects/opencvlibrary/

[17] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”, Communications of the ACM, vol.24, no.6, pp.381–395, 1981.

[18] Open Graphics Library, http://www.opengl.org/

[19] T. Möller and B. Trumbore, “Fast, minimum storage ray-triangle intersection,” Journal of graphics tools, vol.2, no.1, pp.21–28, 1997.

(2013年12月9日受付)

[著者紹介]

本田 俊博 (学生会員)

2012年慶應義塾大学理工学部情報工学科卒．現在，同大学大学院修士課程在学中．スマートフォンを用いた隠消現実感表示システムの研究に従事．

斎藤 英雄 (正会員)

1987年慶應義塾大学理工学部電気工学科卒業．1992年同大学院理工学研究科博士課程電気工学専攻修了．同年同大学理工学部助手，2006年より同大学理工学部情報工学科教授．1997年から99年までカーネギーメロン大学ロボティクス研究所訪問研究員．コンピュータビジョンや，それを仮想現実感や3次元映像メディア処理などに応用する研究に従事．