

## 【目次】

1. はじめに
  1. 前書き
  2. NaturalPoint について
2. マニュアルの使用方法
  1. クイックスタート
  2. 追加情報
3. お使いになる前に
  1. 最小システム要件
  2. 同梱物
  3. ハードウェアの互換性
  4. ソフトウェアとハードウェアのインストール
    1. ソフトウェアのインストール
  5. カメラの設置
  6. カメラのシンク
4. Point Cloud ソフトウェアの使用
  1. 準備作業
  2. カメラキャリブレーションツール
    1. ライセンス
    2. カメラプレビュー
    3. ワンドキャプチャ
    4. キャリブレーションの計算
    5. グラウンドプレーン
    6. 結果
    7. オプション
5. Point Cloud API の使用
  1. アーキテクチャ
  2. インタフェースのレイアウト
  3. 設計上の留意点
    1. 準備作業
    2. マーカーのトラッキング
    3. カメラの位置の読み取り
    4. コネクションポイント
    5. スレッディングの問題
  4. インタフェース
    1. INPPointCloud
    2. INPPointCloudFrame
    3. INPPointCloudPoint
    4. INPPointCloudCamera
  5. リターンコード
6. Rigid Body ソフトウェアの使用
  1. 準備作業
  2. リジッドボディの使用
    1. リジッドボディを作成する

2. リジッドボディの定義と削除
    3. リジッドボディの編集
    4. リジッドボディのロードと保存
  3. トラッキングと記録
    1. トラッキングの状態
    2. リアルタイムトラッキング
    3. 記録と再生
    4. トラッキングデータのエクスポートとストリーミング
  4. アプリケーションのプリファレンス
7. Rigid Body API の使用
  1. 準備作業
  2. Rigid Body API の呼び出し
    1. Rigid Body の開始と終了
    2. Rigid Body インタフェース
    3. リジッドボディストリーミング
    4. リジッドボディフレーム
    5. リジッドボディ制御
    6. Point Cloud インタフェース
    7. リターンコード処理
8. 使用上のヒント
  1. トラッキング環境
  2. トラッキングマーカー
9. ソフトウェアのアップデート

## 1. はじめに

### 1.1 前書き

本ユーザーマニュアルの情報は、予告なしに変更する場合があります。また、NaturalPoint は、本書の情報に関していかなる責任も負わないものとします。本ユーザーマニュアルに記載されているソフトウェアは、使用許諾契約のもとに提供されており、使用許諾契約の条項に従ってのみ利用できるものとします。

© 2007 NaturalPoint Inc. All rights reserved. 本出版物の全部または一部を、その形式または用いる手段を問わず、書面による明示的な許可なしに複製することはできません。

OptiTrack、Point Cloud toolkit、Rigid Body toolkit、NaturalPoint は、NaturalPoint Inc. の商標です。Windows は Microsoft の商標です。本書に記載されているその他の商標はすべて、それぞれの所有者に帰属します。

VRPN の提供に関して、NaturalPoint は、NIH 国立研究資源センターおよび NIH 国立画像生物医学・生物学研究所により支援を受ける、ノースカロライナ大学チャペルヒル校における分子グラフィクスおよび顕微鏡検査に関する NIH National Research Resource に感謝します。

### 1.2 NaturalPoint について

NaturalPoint Inc. は、高品質な光学トラッキング製品を皆様にご提供できることを光栄に存じます。どうぞ、NaturalPoint Tracking Toolkit をお役立てください。

NaturalPoint  
33872 SE Eastgate Circle  
Corvallis, OR 97333  
Telephone: 541-753-6645  
Fax: 541-753-6689  
[www.naturalpoint.com](http://www.naturalpoint.com)

## 2. マニュアルの使用方法

### クイックスタート

Tracking Toolkit の光学トラッキング製品をお使いになる前に、このマニュアルをお読みになることを強くお勧めします。マニュアルをすべて読みとおさずに、製品をできるだけ早く使い始めたい場合には、以下の手順に従ってください。最小の手順で使い始めることができます。

- ◇ インストールのセクションを読み、記述されている手順を実行します。これを行わないと、Tracking Toolkit (Point Cloud および Rigid Body) は動作しません。

- ◇ ソフトウェアをインストールしたら、コンピュータの USB ポートに OptiTrack カメラを接続します。
- ◇ Point Cloud Toolkit のライセンスを有効にします（アクティベート）。
- ◇ デスクトップのショートカットから、OptiTrack Camera Calibration Tool を起動します。

## 追加情報

Tracking Toolkit (Point Cloud および Rigid Body) は、強力な光学トラッキングソリューションで、最高のパフォーマンスを引き出すために設定できる、数多くのオプションが用意されています。製品の動作をより深く理解するためには、Section 4 (「Point Cloud ソフトウェアの使用」)、Section 5 (「Point Cloud API の使用」)、Section 6 (「Rigid Body ソフトウェアの使用」)、Section 7 (「Rigid Body API の使用」) をお読みになることをお勧めします。

## 3. お使いになる前に

### 3.1 最小システム要件

- ◇ 3 台以上の OPTITRACK カメラ
- ◇ Windows XP SP2 または Windows Vista
- ◇ .Net 3.0 RUNTIME
- ◇ Microsoft Visual C++ 2005 RUNTIME
- ◇ 1.5 GHZ のプロセッサ
- ◇ 256 MB の RAM
- ◇ 20 MB のハードディスク空き領域
- ◇ 高速 USB 2.0 ポート

## 3.2 同梱物



Point Cloud または Rigid Body ライセンスカード



クイックスタートガイド



ソフトウェア CD

## 3.3 ハードウェアの互換性

Tracking Toolkit (Point Cloud および Rigid Body) ソフトウェアを使用できるハードウェアは、OptiTrack FLEX:C120、OptiTrack FLEX:V100、OptiTrack SLIM:V100 のみです。これ以前のモデルのカメラは互換性がなく、このソフトウェアでは動作しません。

## 3.4 ソフトウェアとハードウェアのインストール

最良の結果を得るためには、OptiTrack カメラを接続する前にすべてのソフトウェアをインストールすることをお勧めします。

### 3.4.1 ソフトウェアのインストール

注：Windows XP と Vista をお使いの場合は、管理者としてログインする必要があります。コンピュータにユーザーが 1 つしか作成されていない場合は、多くの場合、そのユーザーに管理者権限があります。

同梱の NaturalPoint ソフトウェア CD を、CD ドライブに挿入します。しばらくすると、インストールプログラムが自動的に起動します。数分経ってもインストールプログラムが起動しない場合には、[マイコンピュータ]を開いて CD ドライブのアイコンをダブルクリックし、Setup ファイルをダブルクリックします。

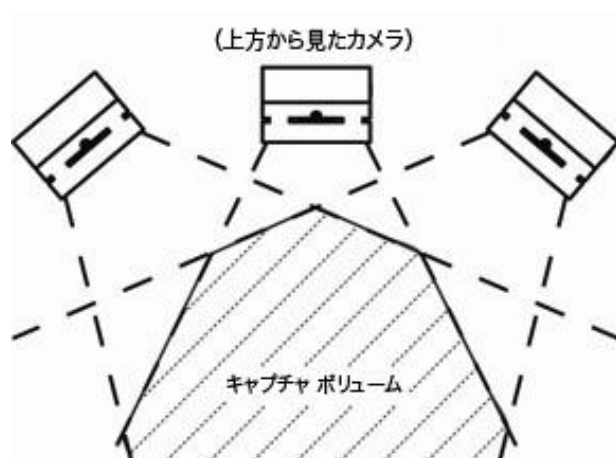
OptiTrack SDK と Tracking Toolkits の 2 つのソフトウェアパッケージをインストールします。最初に OptiTrack SDK のインストーラが起動します。それが完了すると、自動的に Tracking Toolkits のインストーラが起動します。注：コンピュータ上に必要な依存システム (.Net 3.0 など) が存在しない場合は、Tracking Toolkit はそれを検知して、インストールします。

画面上に表示される、ソフトウェアのインストール手順に従います。注：Windows では、ドライバに署名がないことに関する警告メッセージが表示される場合があります。この場合には、[YES] をクリックして、ドライバのインストールを許可します。このドライバはシステムに障害を与えることはありません。

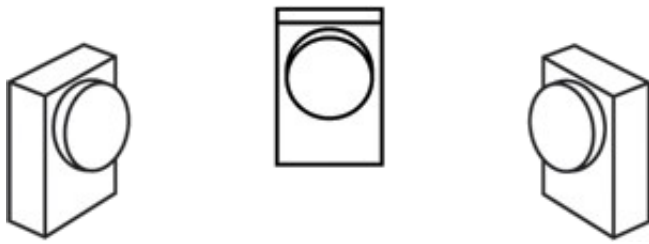
OptiTrack SDK、Point Cloud、Rigid Body のアイコンがデスクトップに表示されます。

### 3.5 カメラの設置

マーカをトラッキングするためには、複数台の OptiTrack カメラを視野が重複するように配置する必要があります。こうすると、「キャプチャボリューム」と呼ばれる、トラッキング可能な領域が作成されます。カメラはできるだけしっかり固定してください。カメラが動いてしまうと、キャリブレーションをやり直さなくてはなりません。



良好なキャリブレーションとトラッキング結果を得るためには、すべてのカメラを同一平面上に設置しないでください。カメラを同一方向に向けるのではなく、別々のアングルで設置します。



### 3.6 カメラのシンク

複数の OptiTrack カメラをリンクして、露出のタイミングをシンクさせることができます。カメラのシンクによって、よりよい精度が得られ、より安定したトラッキングが可能になります。カメラは、シンクケーブル\*を使用してチェーン状に接続します。ケーブルを接続してカメラを起動すると、自動的にシンクが開始されます。

\*シンクケーブルは別途ご購入いただく必要があります。

**注：すべてのカメラは同じモデルでなくてはなりません。モデルが異なると、シンクはうまく動作しません。異なるモデルでのシンクを試行しないでください。**

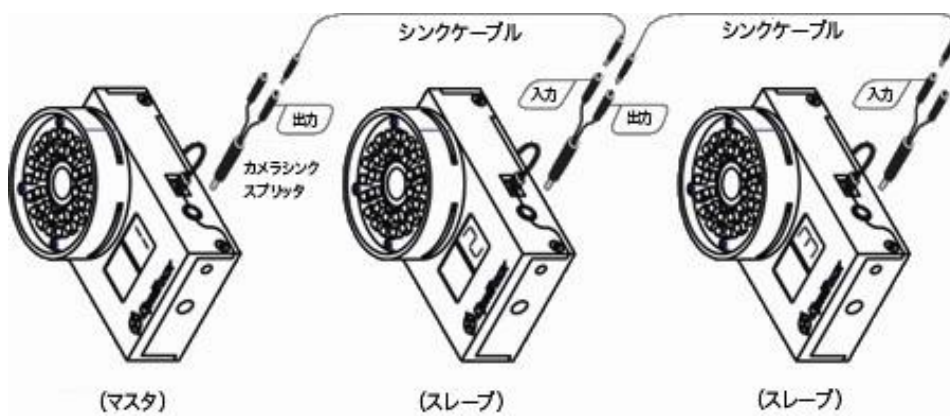
すべてのカメラを USB 経由でコンピュータに接続します。

各カメラにカメラシンクスプリッタを差し込みます。

各カメラの「出力」コネクタと次のカメラの「入力」コネクタシンクケーブルで、チェーン状に接続します。

カメラをループさせないでください。チェーンの最後のカメラを最初のカメラには接続しないでください。

ソフトウェアからカメラを起動します。カメラは自動的にシンクします。



## 4. Point Cloud ソフトウェアの使用

Point Cloud Toolkit は、高度なトラッキングの要求に応える強力なソリューションです。コンパクトなソフトウェアツールキットで複数台の OptiTrack カメラを使用した、3D トラッキングとキャリブレーションが可能です。また、すべてに使いやすい API が付属しています。Point Cloud Toolkit を適切にインストールしてキャリブレーションすれば、お使いのアプリケーションから、安定して正確なトラッキングデータに簡単にアクセスできるようになります。OptiTrack カメラと Point Cloud ソフトウェアから最良の結果を得るために、以下の情報をお役立てください。

### 4.1 準備作業

Point Cloud ソフトウェアを使用する前に、第 3 章に述べた通りにすべてのソフトウェアとハードウェアが適切にインストールされ、ライセンスが有効になっていることを確認してください。インストールが完了していれば、以下の手順に従って Point Cloud ソフトウェアを使い始めることができます。

3 台以上のカメラをコンピュータの USB ポートに接続し、カメラを配置してキャプチャボリュームを設定します (Section 3.5 「カメラの設置」参照)。

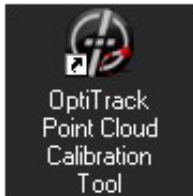
キャリブレーションウィザードを実行してカメラをキャリブレーションし、キャプチャボリュームをテストします。

後にソフトウェアで使用するために、キャリブレーション結果を保存します。

### 4.2 カメラキャリブレーションツール

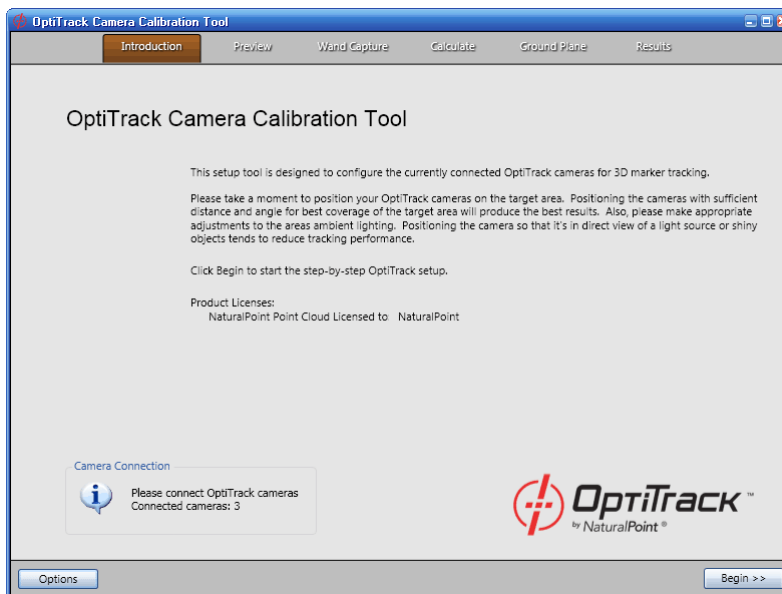
Point Cloud ソフトウェアでマーカーのトラッキングを行う前に、カメラのキャリブレーションを実行する必要があります。キャリブレーションによって、OptiTrack カメラの設定を微調整し、カメラの物理的位置 (外的パラメータ) とカメラのレンズのプロパティ (内的パラメータ) を確認することができます。また、キャリブレーションの手順をガイドするキャリブレーションツールでは、結果を確認したり、サンプルデータをキャプチャする方法も表示されます。キャリブレーションをはじめる前に、カメラをセットアップする必要があります (Section 3.5 「カメラの設置」参照)。たとえ 1 台でもカメラの位置を動かしてしまうと、新たにキャリブレーションを行う必要があります。

次のセクションでは、キャリブレーションツールの使用方法について述べます。ツールを起動するには、デスクトップの [OptiTrack Point Cloud Calibration Tool] のアイコンをクリックします。



#### 4.2.1 ライセンス

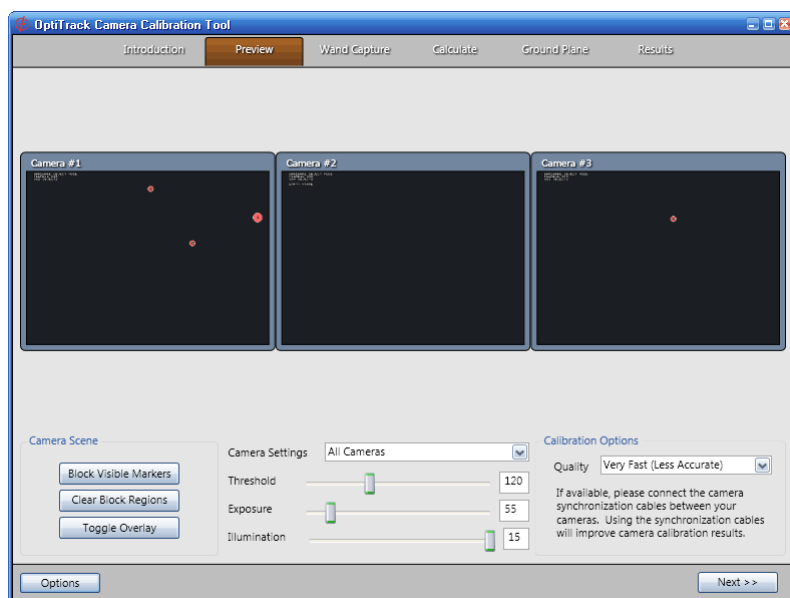
カメラキャリブレーションの最初の手順では、有効なライセンスと適合したカメラがあることが確認されます。また、3台以上のカメラがコンピュータに接続されていることも確認されます。カメラが3台以上接続されていないと、キャリブレーションおよびトラッキングを行うことはできません。



すべての準備が整ったら、[Begin]（開始）ボタンを押して次の手順に進みます。

#### 4.2.2 カメラプレビュー

カメラプレビューの手順では、カメラの視点からのシーンが表示されます。この手順の目的は、カメラが感知してしまう不要な光をすべて除去することです。このためには、障害になるものを片付け、ライトを消し、窓を覆い、カメラ設定を調整し、ブロッキングツールを使用します。このプロセスで、キャプチャボリュームにグラウンドプレーンとワンドが入っていないことを確認してください。



## カメラの設定

OptiTrack カメラでは、設定を最適化してシーンの不要な光を低減することができます。設定を調整する際には、個別のカメラに対して設定を変更することも、[Camera Settings]（カメラ設定）ドロップダウンリストを使って一度にすべてのカメラの設定を変えることもできます。

- ◇ **Threshold（しきい値）**：このスクロールバーでは、カメラがマーカを認識するためには、マーカがどの程度明るくならないかを調整します。「180」のような大きい値を使うと、シーン内の不要な要素の数を減らすことができます。高すぎる値（「230」以上）を使うと、検出されたマーカの精度が悪化する場合があります。
- ◇ **Exposure（露出）**：このスクロールバーでは、カメラのシャッタータイマーを調整して、各フレームでカメラが取り込む光の量を決定します。小さな値（「55」など）を設定すると、シーンで検出される不要な要素の数を減らすことができます。
- ◇ **Illumination（イルミネーション）**：このスクロールバーでは、OptiTrack カメラに搭載された IR LED が発する光源の明るさを調節します。

OptiTrack FLEX:V100 カメラを使って最良のパフォーマンスを引き出すためには、IR ストロボモードを使用します。[Illumination]の値を「15」に設定すると、このモードが有効になります。ストロボモードが有効の場合は、[Exposure]の値を「100」以上に設定しても意味がありません。その露出時間以降は LED が発光しないためです。ストロボモードで LED の明るさを調節するには、[Exposure]の設定を「1」から「100」までの範囲（小さい値は光量が少なくなります）で変更します。

## ブロッキングツール

ブロッキング領域を使用すると、カメラの視野内で検出されたライト（マーカ、光源など）を無視するように設定することができます。ブロッキング領域は、カメラプレビューにマウスで四角形を描いて設定したり、[Block Visible Markers]（可視マーカをブロック）ボタ

ンで自動的に設定することができます。すべてのブロック領域を除去するには、[Clear Block Regions]（ブロック領域をクリア）ボタンを押します。

#### キャリブレーションのオプション

キャリブレーション結果の品質は、時間をかけて調整を行うことによって向上させることができます。通常は、手早く設定しても十分な品質が得られます。キャリブレーションレベルを選択するには、[Quality]（品質）ドロップダウンリストを使用します。

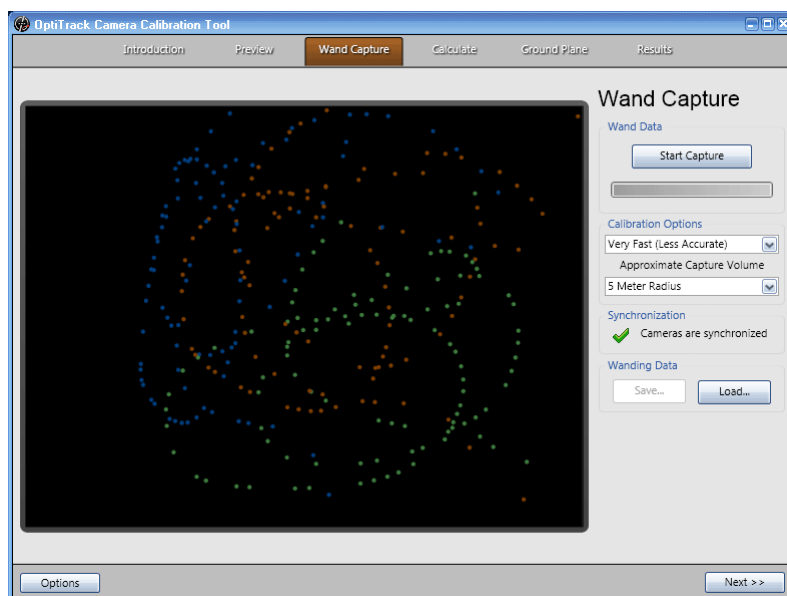
シーンがクリアされ、すべてのマーカーが除去されるかブロックされるかしたら、[Next]（次へ）ボタンを押します。

#### 4.2.3 ワンドキャプチャ

ワンドキャプチャの手順は、カメラの物理的な位置とレンズの特性を計算するために使用するデータセットを、カメラから収集することが目的です。ソフトウェアが記録している間は、ワンド（取っ手付きの反射マーカー）を、ゆっくりと立体的に、キャプチャボリューム全体に渡って動かす必要があります。ワンドをシーン全体に渡って動かすと、画面上には、各カメラについて異なる色の円が表示されます。サンプル点が取得されると、やや暗い色で画面に表示されます。

各カメラの撮影範囲をまんべんなくサンプリングするよう、特に注意を払ってください。記録したサンプル点は、視覚化情報およびワンドの範囲の目安として使用することができます。ワンドを円状に立体的なパターンで動かすと、良好な結果が得られます。また、動きの深度を生み出すために、ワンドを頻繁にカメラから放したり近づけたりするようにしてください。

すべてのカメラにとって、シーンで唯一検出するマーカーがワンドであることが重要です。別のマーカーがあると、エラーが発生してキャリブレーションの質が落ちる可能性があります。その際は、ソフトウェアは警告を表示するとともに、余分なマーカーを無視しようとしませんが、余分なマーカーがある場合には、前述の「カメラプレビュー」の手順に戻ってそれらを取り除くことをお勧めします。



## ワンドデータ

ワンドデータのキャプチャプロセスを開始する準備ができたなら、[Start Capture] (開始) ボタンを押します。データが収集されるとともに、プログレスバーが更新されて残り時間が示されます。

## シンク

シンクインジケータによって、OptiTrack カメラがどの程度良好にシンクしているかがわかります。ハードウェアカメラシンクを使うと、キャリブレーションと 3D トラッキングのパフォーマンスを大幅に改善することができます。詳細は、Section 3.5 「カメラのシンク」を参照してください。

ワンドデータを収集したら、[Next] (次へ) ボタンを押します。

### 4.2.4 キャリブレーションの計算

前述の手順でワンドデータのセットをキャプチャしたら、処理を行う必要があります。計算にかかる時間の長さは、「カメラプレビュー」の手順で選択した、キャリブレーションの品質レベルによって異なります。キャリブレーションを開始するために、[Start Calculation] (計算開始) ボタンを押します。

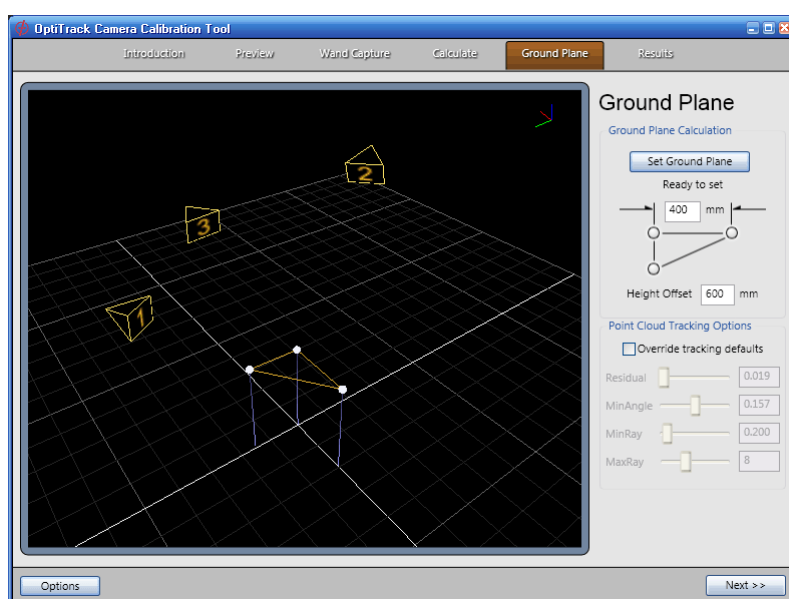
計算が完了すると、スクロールが停止し、得られた品質のレベルに関する情報が表示されます。[Standard Error] (標準誤差)、[Mean Error] (平均誤差) に表示される数値が小さいほど良好です。通常、望ましい誤差は、[Standard Error] が「0.5」以下、[Mean Error] が「0.5」以下です。計算結果が満足のものでない場合は、「ワンドキャプチャ」の段階に戻り、新しくデータセットを収集することを検討します。

良好なキャリブレーション結果が得られたら、[Next] (次へ) ボタンを押します。

#### 4.2.5 グラウンドプレーン

グラウンドプレーンの手順では、物理的な地面（床）の高さを特定し、座標系の尺度と 3D 世界の軸の方向を設定することができます。キャリブレーションの手順で計算した位置を使用して、カメラが 3D で表示されます。また、シーンで検出したマーカの 3D 座標データも再構築され、結果が表示されます。グラウンドプレーンが設定されてはじめて、期待した位置にカメラとマーカが表示されます。

グラウンドプレーンは、3つの頂点に反射マーカがついた 3:4:5 の三角形です。キャプチャボリューム内にグラウンドプレーンを配置すると、間を直線で結んだ状態で、その3つのマーカが 3D ビューに表示されます。三角形が検知されて、シーンにマーカが3つだけ表示されているときに、[Set Ground Plane]（グラウンドプレーンを設定）ボタンを押します。ボタンを押すと、カメラとマーカの場所が更新されます。



#### 3D ビュー

- ◇ **ビューの回転**：マウスの左ボタンを押したままマウスを動かすと、3D ビューを回転することができます。
- ◇ **ビューの移動**：マウス右ボタンを押したままマウスを動かすと、3D ビューを移動することができます。
- ◇ **ビューのズーム**：マウスのホイールを使用するか、マウスの左右ボタンを押したままマウスを動かすと、3D ビューを拡大または縮小することができます。

#### グラウンドプレーンの計算

- ◇ **Set Ground Plane (グラウンドプレーンを設定)**：地面が希望通りの位置に表示されたら、このボタンを押してキャリブレーションを更新します。シーンに3つのマーカが存在する場合にのみボタンを押すことができます。余分なマーカがある場合は、ボタンは無効になります。

- ◇ **Ground Plane Size(グラウンドプレーンのサイズ)**: 地面を表す 3:4:5 の三角形の「4」にあたる辺の長さを、ミリメートル単位で入力します。「4」の辺が Z 軸になります。デフォルトのサイズは 300mm、400mm、500mm です。この数値はすべてのトラッキングデータに対する座標系の尺度に使用されるので、正確に設定することが重要です。この数値は、[Set Ground Plane] (グラウンドプレーンを設定) ボタンを押す前に入力する必要があります。
- ◇ **Height Offset (高さのオフセット)**: グラウンドプレーンが正確に床 (地面) と同じ高さがない場合は、差異を補正するために高さのオフセットをミリメートルで入力することができます。この数値は、[Set Ground Plane] (グラウンドプレーンを設定) ボタンを押す前に入力する必要があります。

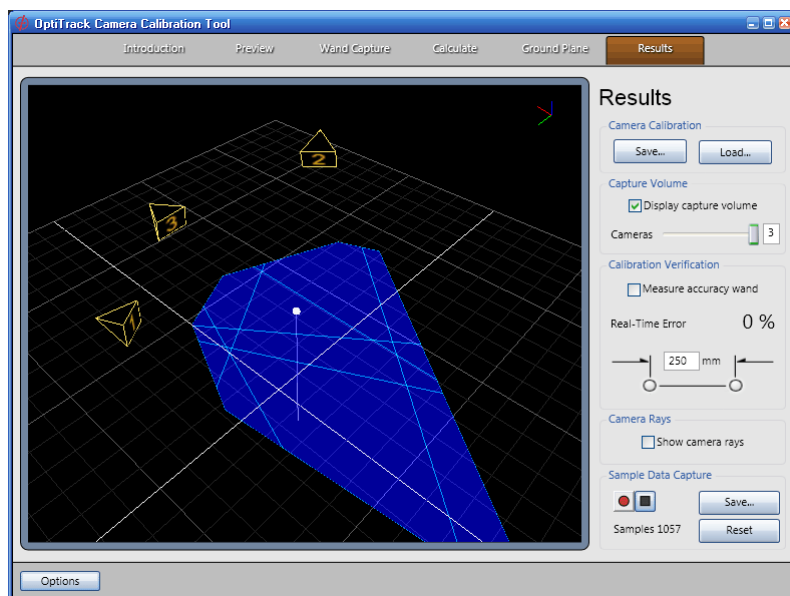
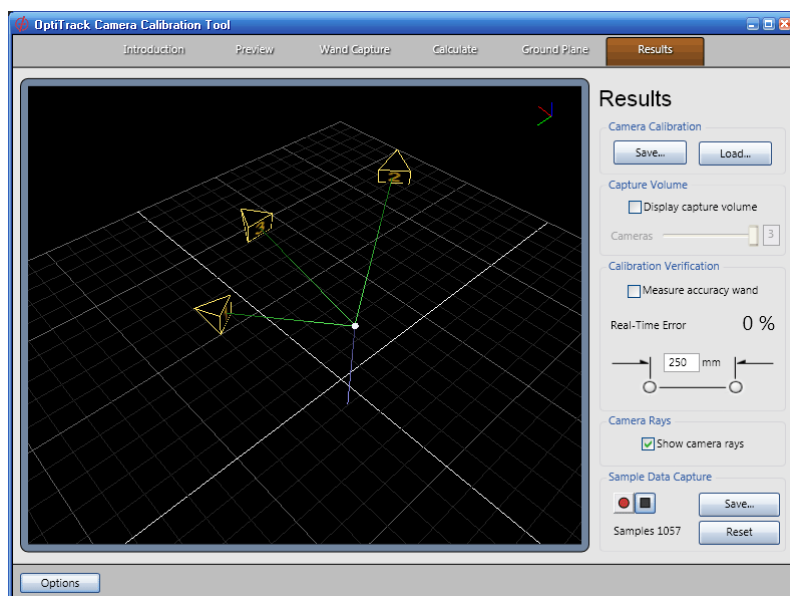
#### Point Cloud トラッキングオプション

2D の位置データから 3D マーカーへの再構築を制御するパラメータは、必要に応じて手動で微調整することができます。通常は、デフォルトの値で良好な結果が得られます。

グラウンドプレーンを設定したら、[Next] (次へ) ボタンを押して次の手順へ進みます。

#### 4.2.6 結果

キャリブレーションのプロセスはこれで完了です。この最後の手順では、キャリブレーションの結果を確認して、後で利用できるように保存します。データを可視化し、品質を計測するためのツールがいくつか提供されています。Point Cloud COM オブジェクトを使用するカスタムアプリケーションがある場合、トラッキングを行うためには、この段階で保存したキャリブレーションファイルをロードする必要があります。また、既存のキャリブレーションをロードして検査したり、利用することもできます。



### カメラのキャリブレーション

後で利用するためにキャリブレーションを保存したり、以前のキャリブレーションを取得したりするためには、[Save]（保存）および[Load]（ロード）ボタンを使用します。

### キャプチャボリューム

このツールによって、キャプチャボリュームの物理的領域を可視化し、良好なトラッキング範囲を決定することができます。全カメラの撮影範囲の領域を表示したり、複数のカメラが重複している部分を表示したりするように調整することができます。

### キャリブレーションの検証

あらかじめ距離がわかっている、2つのマーカーが付いたワンドを使うことによって、キャリブレーションの品質を計測することができます。このワンドをアキュラシー（精度）ワンド

と呼びます。[Measure accuracy wand]（アキュラシーワンドの計測）チェックボックスをオンにして、ワンドをキャプチャボリューム全体に渡って動かすと、ワンドの位置に対するリアルタイムの誤差見積もりが表示されます。ワンドの点の距離はミリメートルで入力する必要があります。

### カメラレイ

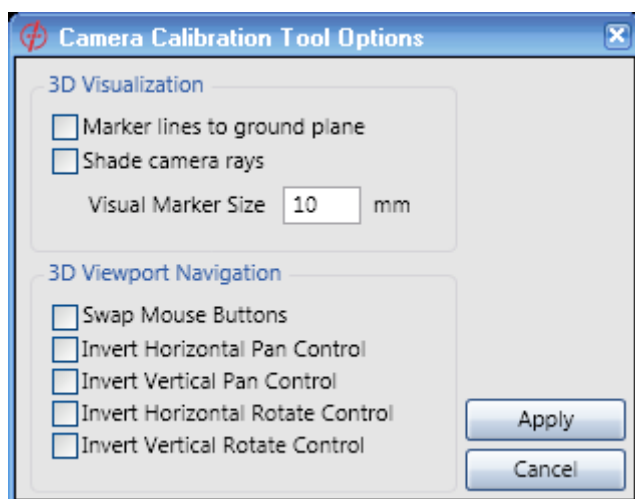
Camera Rays ツールは、どのカメラがどのマーカールを見ることができるかを測定するために使用します。すべてのマーカールと、それを検知しているカメラを直線で結んだ状態で、この結果が示されます。

### サンプルデータのキャプチャ

キャプチャサンプルを記録して保存すると、この領域から、3D マーカーのトラッキングデータを CSV ファイルフォーマットでエクスポートできます。記録を開始するには赤い丸の付いたボタンを使用し、停止するには黒い四角が付いたボタンを押します。[Save]（保存）ボタンを押すと、記録したデータをファイルに保存することができます。エクスポートしたデータは、最初の列にマーカールの数、第2列目にタイムスタンプ、続いてマーカールごとの(x, y, z)を記述した3列の形式で、フォーマットされています。

#### 4.2.7 オプション

キャリブレーションツールのインタフェースのいくつかの機能は、必要に応じてカスタマイズできます。カスタマイズを制御する設定は、[Options]（オプション）ウィンドウにあります（キャリブレーションツール左下の[Options]（オプション）ボタンで起動）。これらの設定への変更は、キャリブレーションツールの終了時に保存され、再開すると復元されます。

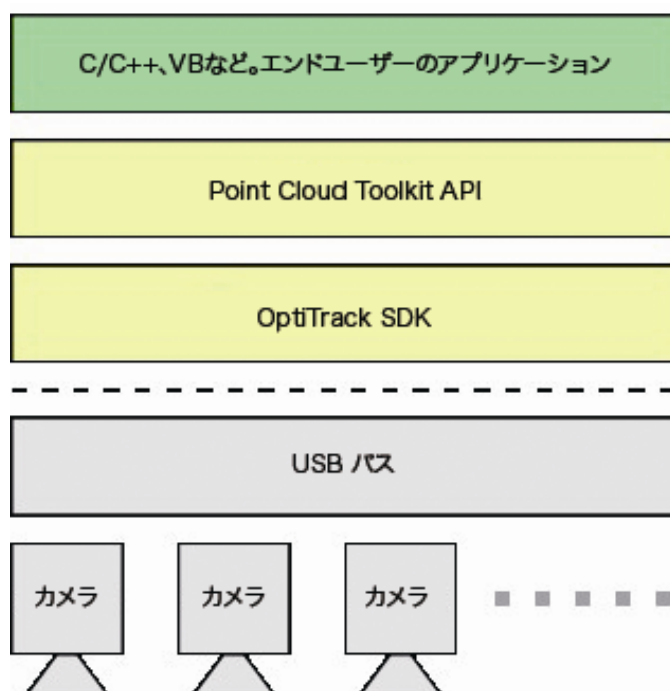


## 5. Point Cloud API の使用

アプリケーションで Point Cloud トラッキングを使用するためには、Point Cloud API を使ってそれに対するサポートを実装する必要があります。次のセクションでは、システムのアーキテクチャの概要と API に関する仕様の詳細を説明します。

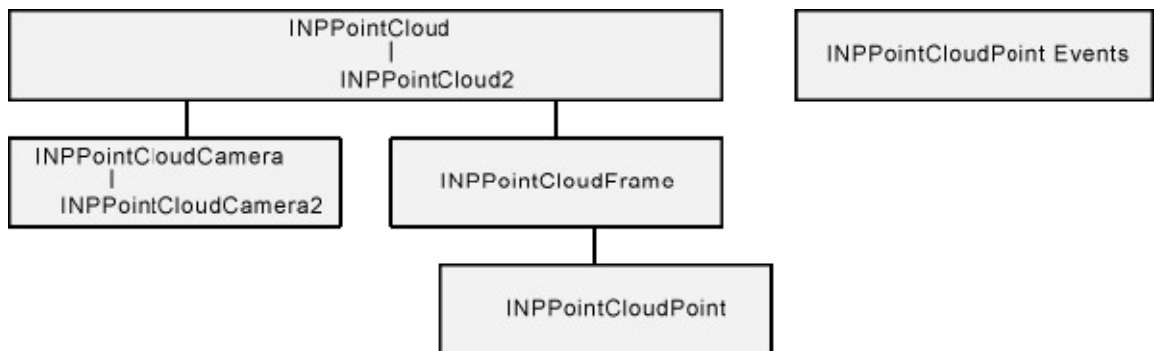
### 5.1 アーキテクチャ

Point Cloud API は、一連の COM オートメーションインタフェースとして記述されており、OptiTrack SDK 上で構築されています。このタイプのインタフェースを選択したのは、高い柔軟性を備えているという理由からです。COM オートメーションインタフェースは、VBScript、JavaScript、Visual Basic、C/C++を含むほぼすべての言語でサポートされています。Python、Delphi など、他の多くの言語に対するサポートもしています。詳細は任意の言語に関する記述を参照してください。



### 5.2 インタフェースのレイアウト

下の図は、Point Cloud API のインタフェースとそれらの関係を示しています。



メインのインタフェースは INPPointCloud です。このインタフェースによって、カメラを列挙し、キャリブレーションプロファイルをロードし、トラッキングデータを処理することができます。3D データのフレームの準備ができると、クライアントは INPPointCloudFrame オブジェクトを取得し、そのフレームで検知されたマーカが INPPointCloudPoint 経由で使用できます。3D データの新しいフレームが使用可能となる時期に関する通知は、INPPointCloudEvents インタフェースを介して受信されます。

### 5.3 設計上の留意点

#### 5.3.1 準備作業

ほんの少量のコードを書くだけで、使いはじめることができます。カメラの初期化手順を以下に概説します。

ファイルに保存されている結果を使用してカメラがキャリブレーションされていることを確認します (Section 4.2 「カメラキャリブレーションツール」参照)。

INPPointCloud オブジェクトを作成します。

INPPointCloud->LoadProfile() メソッドを使用して、既存のキャリブレーションプロファイルをロードします。

INPPointCloud->Start() メソッドを呼び出し、カメラを列挙してから起動します。

この時点で、カメラは初期化されてフレーム情報を収集しているはずですが、アプリケーションのメインループでは、INPPointCloudEvents インタフェースを使ってフレームコールバックを扱うか、INPPointCloud->GetFrame() を使ってフレームをポーリングする必要があります。

データ処理が完了したら、INPPointCloud->Stop() メソッドを呼び出します。

#### 5.3.2 マーカーのトラッキング

Point Cloud API は、複数の OptiTrack カメラから 2D トラッキングデータを収集し、それを 3D の位置データとして結合するという、強力な機能への単純なインタフェースを提供します。

3D データの単一のフレームを作成するため、Point Cloud ソフトウェアは存在する各カメラから1つずつフレームを収集します。すべてのフレームを収集して 3D 位置データが計算されると、クライアントアプリケーションに通知するために、OnFrameAvailable() コールバックが発生します。また、クライアントは、INPPointCloud->GetFrame() を使用して新しいフレームをポーリングすることもできます。次に、クライアントはフレームに対する INPPointCloudFrame オブジェクトを取得します。

こうすると、Item() 呼び出しを使用して、個別のマーカーを INPPointCloudPoints として列挙することができます。クライアントアプリケーションで使用するため、マーカーの X、Y、Z 軸上の位置がこの時点で抽出されます。

### フレームオブジェクトの寿命

Point Cloud システムにおけるフレームオブジェクトは限りのあるリソースです。Free() メソッドを必ず呼び、できるだけ早く INPPointCloudFrame インタフェースを解放してください。

### 座標系

すべての位置データはメートル単位で返されます。Section 4.2「カメラキャリブレーションツール」で述べたように、座標系の絶対精度は、良好なキャリブレーションとグラウンドプレーンの正確な使用に依存しています。

#### 5.3.3 カメラの位置の読み取り

キャリブレーションがロードされると、キャリブレーション実行時に接続されていた、すべてのカメラの位置と方向が認識されます。位置データは、INPPointCloud インタフェースの GetCamera() 呼び出しでカメラを列挙することによって抽出できます。これは、X、Y、Z と回転のデータを読み取るために使用できる、INPPointCloudCamera オブジェクトを返します。

#### 5.3.4 コネクションポイント

コールバック通知には、標準の COM コネクションポイントが使用されます。コネクションポイントを選択した理由は、COM オートメーションインタフェースと通信する、ほとんどの言語と互換性があるためです。詳しくは MSDN ヘルプの COM 関連文書を参照してください。特に、IConnectionPoint と IConnectionPointContainer をお読みください。C/C++ではコネクションポイントの設定にやや手間がかかりますが、コーディングを支援するためのサンプルコードが提供されています。

#### 5.3.5 スレディングの問題

OptiTrack API はアパートメントスレッドです。アパートメントスレッドとは、任意の時間に DLL へのコールをするためにただ1つのスレッドしか使用できないことを意味します。あるスレッドが DLL で実行中の場合には、その DLL への呼び出しに使われる他のすべてのスレッドは、最初のスレッドが完了するまでブロックされます。これはコールバックの実装方法にも影響を与えます。コネクションポイントのコールバックは、オブジェクトを作成した同じスレッドで実行する必要があります。コネクションポイントを持つオブジェクトの作成時に、隠しウィンドウを作成することによって、これを実行することができます。隠しウィンドウは、呼び出しスレッドのメッセージキューを使います。コールバックは、隠しウィンドウへメッセージをポストして、コネクションポイントコールバックインタフェースを呼び出すこ

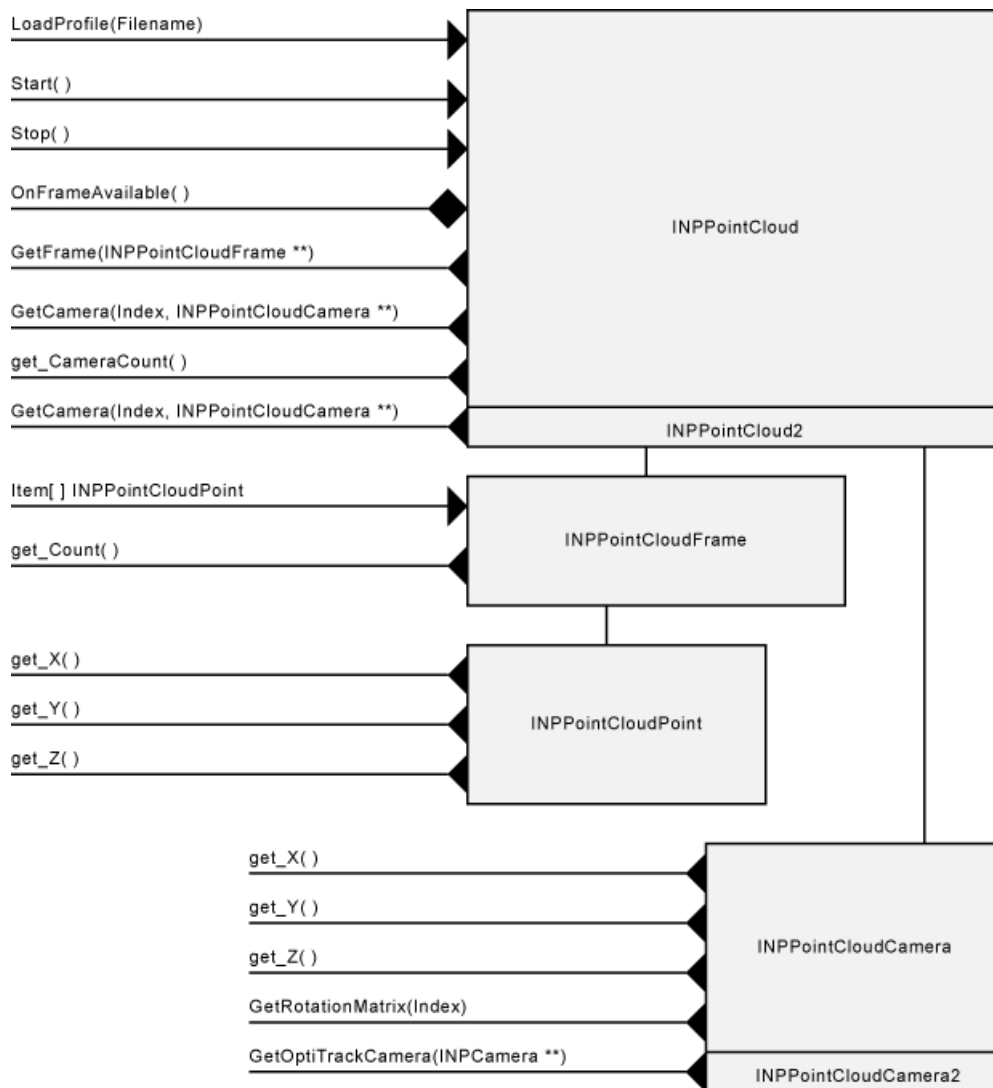
とによって実行されます。このメカニズムの欠点は、OptiTrack 通知をタイムリーに処理するためには、これを実行するために、オブジェクトを作成したスレッドに対するメッセージキューを解放する必要があることです。たとえば、OptiTrack カメラオブジェクトを作成するために GUI アプリケーションのメインのスレッドを使う場合は、そのスレッドでブロック操作が実行されていないことを確実にします。ブロック操作が必要な場合は、メッセージの処理にはメッセージループを使用します。

この問題は、コネクションポイントコールバックにのみ影響を及ぼします。この INPPointCloud インタフェースのポーリングによってカメラのフレーム情報を収集する場合には、これは問題にはなりません。

## 5.4 インタフェース

### 5.4.1 INPPointCloud

Point Cloud Toolkit と通信するためのメインのエントリポイントです。このインタフェースによって、カメラを列挙し、キャリブレーションプロファイルをロードし、トラッキングデータを処理することができます。



#### 5.4.1.1 INPointCloud::LoadProfile()

LoadProfile()メソッドは、ファイルに保存されたキャリブレーションプロファイルのロードを試行します。プロファイルへのパスを取り、成功したかどうかの情報を返します。

```
HRESULT LoadProfile(BSTR FileName);
```

#### パラメータ

- FileName  
[in]キャリブレーションプロファイルの場所へのパス

#### パラメータ

値	意味
S_OK	メソッドは成功
NP_POINTCLOUD_LOAD_FAILED	メソッドは失敗、プロファイルはロードされない

#### 5.4.1.2 INPointCloud::Start()

Start()メソッドは、トラッキングを開始するためにカメラを列挙し、起動します。このメソッドはLoadProfile()メソッドの後に呼び出す必要があります。

```
HRESULT Start();
```

#### パラメータ

- なし

#### パラメータ

値	意味
S_OK	メソッドは成功
NP_POINTCLOUD_FAILED	メソッドは失敗、カメラは起動されない

#### 5.4.1.3 INPointCloud::Stop()

Stop()メソッドは、トラッキングを停止するためにカメラを停止します。

```
HRESULT Stop();
```

#### パラメータ

- なし

#### パラメータ

値	意味
S_OK	メソッドは成功
NP_POINTCLOUD_FAILED	メソッドは失敗、カメラは停止しない

#### 5.4.1.4 INPointCloud::GetFrame()

GetFrame()メソッドは、処理済みの3Dトラッキングデータのフレームを含むINPointCloudFrameオブジェクトを取得します。

```
HRESULT GetFrame(INPointCloudFrame **NPointCloudFrame);
```

#### パラメータ

- INPPointCloudFrame  
[out] INPPointCloudFrame インタフェースを受け取るポインタです。フレームがない場合は、NULL が戻されます。

#### パラメータ

値	意味
S_OK	メソッドは成功

#### 5.4.1.5 INPPointCloud::GetCamera()

GetCamera() メソッドは、接続された OptiTrack カメラの 1 台について、INPPointCloudCamera オブジェクトを取得します。

```
HRESULT GetCamera([in] LONG Index, [out] INPPointCloudCamera  
**NPPointCloudCamera);
```

#### パラメータ

- Index  
[in] 取得するアイテムのインデックスです。
- NPPointCloudCamera  
[out] INPPointCloudCamera インタフェースを受け取るポインタです。カメラがない場合は、NULL が戻されます。

#### パラメータ

値	意味
S_OK	メソッドは成功

#### 5.4.1.6 INPPointCloud::get\_CameraCount()

get\_CameraCount() メソッドは、接続されている OptiTrack カメラの数を取得します。

```
HRESULT get_CameraCount([out, retval] LONG *pVal);
```

#### パラメータ

- pVal  
[out] 接続されているカメラの数を LONG 型で受け取るポインタです。

#### パラメータ

値	意味
S_OK	メソッドは成功

#### 5.4.1.7 INPointCloud2::GetCamera()

INPointCloud2 GetCamera() メソッドは、接続された OptiTrack カメラの 1 台について、INPointCloudCamera2 オブジェクトを取得します。このメソッドの INPointCloud2 バージョンは、INPointCloud とは異なる戻り値を使います。

```
HRESULT GetCamera([in] LONG Index, [out, retval] INPointCloudCamera2  
**NPointCloudCamera);
```

#### パラメータ

- ・ Index  
[in] 取得するアイテムのインデックスです。
- ・ NPointCloudCamera2  
[out, retval] INPointCloudCamera2 インタフェースを受け取るポインタです。カメラがない場合は、NULL が戻されます。

#### パラメータ

値	意味
S_OK	メソッドは成功

#### 5.4.2 INPointCloudFrame

このオブジェクトは、処理済みの 3D トラッキングデータの現在のフレームに関する情報を保持します。このインタフェースは、フレーム内のすべてのオブジェクトのリストを保持する、標準的な COM の列挙型インタフェースです。

#### 5.4.2.1 INPointCloudFrame::get\_Count()

get\_Count() メソッドは、フレーム内で検出された 3D マーカーの数を取得します。

```
HRESULT get_Count([out, retval] LONG *pVal);
```

#### パラメータ

- ・ pVal  
[out] フレーム内の 3D マーカーの数を LONG 型で受け取るポインタです。

#### パラメータ

値	意味
S_OK	メソッドは成功

#### 5.4.2.2 INPointCloudFrame::Item()

Item()によって、フレームで検出された 3D マーカーを列挙することができます。

```
HRESULT Item([in]LONG a_vlIndex,[out, retval]INPointCloudPoint  
**NPointCloudPoint);
```

#### パラメータ

- ・ Index  
[in]取得するアイテムのインデックスです。
- ・ NPointCloudPoint  
[out, retval] INPointCloudPoint インタフェースを受け取るポインタです。  
マーカーがない場合は、NULL が戻されます。

#### パラメータ

値	意味
S_OK	メソッドは成功

#### 5.4.3 INPointCloudPoint

このオブジェクトは、フレームにおいて検出された単一のマーカーに関する、3D の位置情報 (X, Y, Z) を保持します。

#### 5.4.3.1 INPointCloudPoint::get\_X()

get\_X() メソッドは、フレームで検出された 3D マーカーの X 軸上の位置を取得します。

```
HRESULT get_X([out, retval] DOUBLE *pVal);
```

#### パラメータ

- ・ pVal  
[out]3D マーカーの X 軸上の位置を Double 型で受け取るポインタです (メートル単位)。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.3.2 INPointCloudPoint::get\_Y()

get\_Y() メソッドは、フレームで検出された 3D マーカーの Y 軸上の位置を取得します。

```
HRESULT get_Y([out, retval] DOUBLE *pVal);
```

#### パラメータ

- ・ pVal  
[out] 3D マーカーの Y 軸上の位置を Double 型で受け取るポインタです（メートル単位）。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.3.3 INPointCloudPoint::get\_Z()

get\_Z() メソッドは、フレームで検出された 3D マーカーの Z 軸上の位置を取得します。

```
HRESULT get_Z([out, retval] DOUBLE *pVal);
```

#### パラメータ

- ・ pVal  
[out] 3D マーカーの Z 軸上の位置を Double 型で受け取るポインタです（メートル単位）。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.4 INPointCloudCamera

このオブジェクトは、接続された OptiTrack カメラに関する、3D 位置 (X, Y, Z) と方向（回転）の情報を保持します。

#### 5.4.4.1 INPointCloudCamera::get\_X()

get\_X() メソッドは、カメラの X 軸上の位置を取得します。

```
HRESULT get_X([out, retval] DOUBLE *pVal);
```

#### パラメータ

- ・ pVal  
[out]カメラのX軸上の位置を Double 型で受け取るポインタです（メートル単位）。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.4.2 INPointCloudCamera::get\_Y()

get\_Y() メソッドは、カメラのY軸上の位置を取得します。

```
HRESULT get_Y([out, retval] DOUBLE *pVal);
```

#### パラメータ

- ・ pVal  
[out]カメラのY軸上の位置を Double 型で受け取るポインタです（メートル単位）。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.4.3 INPointCloudCamera::get\_Z()

get\_Z() メソッドは、カメラのZ軸上の位置を取得します。

```
HRESULT get_Z([out, retval] DOUBLE *pVal);
```

#### パラメータ

- ・ pVal  
[out]カメラのZ軸上の位置を Double 型で受け取るポインタです（メートル単位）。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.4.4 INPointCloudCamera::GetRotationMatrix()

GetRotationMatrix()メソッドは、カメラ回転行列の配列から1つの要素を取得します。

```
HRESULT GetRotationMatrix([in] LONG Index, [out, retval] DOUBLE *pVal);
```

#### Parameters

- ・ Index  
[in]取得する、配列内のアイテムのインデックスです。配列には9つの要素があり、有効な入力の範囲は0から9までです。
- ・ pVal  
[out]カメラ回転行列の配列における、選択アイテムのDouble型を受け取るポインタです。

#### パラメータ

値	意味
S_OK	メソッドは成功
E_POINTER	ポインタが無効

#### 5.4.4.5 INPointCloudCamera2::GetOptiTrackCamera()

GetOptiTrackCamera()メソッドは、OptiTrack INPCamera オブジェクトを取得します。これは INPointCloudCamera2 を使用している場合にのみ使用可能です。

```
HRESULT GetOptiTrackCamera([out, retval] INPCamera **NPCamera);
```

#### パラメータ

- ◆ Index
- ◆ pVal

[out, retval]OptiTrack INPCamera インタフェースを受け取るポインタです。カメラがない場合は、NULL が戻されます。

## パラメータ

値	意味
S_OK	メソッドは成功

### 5.5 リターンコード

Point Cloud API の失敗時のリターンコードのリストは以下の通りです。

```
enum __MIDL__MIDL_itf_PointCloudCOM_0000_0001
{
    NP_POINTCLOUD_FILE_NOT_FOUND = 1,
    NP_POINTCLOUD_LOAD_FAILED = NP_POINTCLOUD_FILE_NOT_FOUND + 1,
    NP_POINTCLOUD_FAILED = NP_POINTCLOUD_LOAD_FAILED + 1,
    NP_POINTCLOUD_CAMERAS_ALREADY_STARTED = NP_POINTCLOUD_FAILED + 1,
    NP_POINTCLOUD_CAMERAS_NOT_RUNNING = NP_POINTCLOUD_CAMERAS_ALREADY_STARTED + 1,
    NP_POINTCLOUD_PROFILE_ALREADY_LOADED = NP_POINTCLOUD_CAMERAS_NOT_RUNNING + 1,
    NP_POINTCLOUD_CAMERAS_RUNNING = NP_POINTCLOUD_PROFILE_ALREADY_LOADED + 1,
    NP_POINTCLOUD_INVALID_FILE = NP_POINTCLOUD_CAMERAS_RUNNING + 1,
    NP_POINTCLOUD_INVALID_PROFILE_CAMERA_COUNT = NP_POINTCLOUD_INVALID_FILE + 1,
    NP_POINTCLOUD_UNABLE_TO_INITIALIZE_OPTITRACK_CAMERAS =
        NP_POINTCLOUD_INVALID_PROFILE_CAMERA_COUNT + 1,
    NP_POINTCLOUD_FAILED_LICENSE_CHECK =
        NP_POINTCLOUD_UNABLE_TO_INITIALIZE_OPTITRACK_CAMERAS + 1,
    NP_POINTCLOUD_NO_PROFILE_LOADED = NP_POINTCLOUD_FAILED_LICENSE_CHECK + 1,
    NP_POINTCLOUD;
}
```

## 6. Rigid Body ソフトウェアの使用

Rigid Body Toolkit は、安定した、リアルタイム 3D 光学トラッキングソリューションです。マーカーは、既知のパターンの複数のオブジェクト（リジッドボディ）にアタッチすることができ、6つのパラメータ（位置と方向）のトラッキングが可能です。トラッキングデータには、ネットワークストリーミングサポートによって、または使いやすいソフトウェア API によって、リアルタイムでアクセスできます。OptiTrack カメラと Rigid Body ソフトウェアから最良の結果を引き出すために、以下の情報を役立ててください。

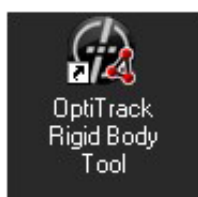
### 6.1 準備作業

Rigid Body ソフトウェアを使用する前に、第 3 章の説明に従って、すべてのソフトウェアとハードウェアが適切にインストールされ、ライセンスが有効化されていることを確認します。インストールが完了したら、以下の手順を使用して、Rigid Body ソフトウェアを使い始めることができます。

3 台以上のカメラをコンピュータの USB ポートに接続し、カメラを調整してキャプチャボリュームを設定します (Section 3.5 「カメラの設置」 参照)。

Point Cloud Calibration ツールでカメラをキャリブレーションし、キャリブレーションをファイルに保存します (Section 4.2 「カメラキャリブレーションツール」 参照)。

デスクトップの [OptiTrack Rigid Body Tool] アイコンをクリックして、Rigid Body ソフトウェアを起動します。



Rigid Body ソフトウェアにキャリブレーションをロードして、カメラから 3D 点群データのキャプチャを開始します。

トラッキングするリジッドボディは、3D 点群データから、あるいは既存のリジッドボディの定義をファイルからロードして定義できます。

### 3D ビューポートの制御

ソフトウェアの 3D ビューポートは、カメラ、3D 点群データ、トラッキングするリジッドボディを表示するために使います。また、リジッドボディの定義の際には、マーカーを選択するためにも使用します。マーカーの選択と 3D ビューポートの操作には、マウスを使用します。

- ◆ マウスの左ボタンは、マーカーの選択に使用します。
- ◆ マウスの右ボタンを押したままマウスを動かすと、3D ビューが回転します。
- ◆ マウスのスクロールホイールは、3D ビューを拡大／縮小します。
- ◆ マウスの中ボタンを押したままマウスを動かすと、3D ビューがパン／移動します。

## 6.2 リジッドボディの使用

OptiTrack Rigid Body ソフトウェアは、キャプチャボリューム内のカメラに対して可視の反射マーカーから、3D 点の集合を作成します。リジッドボディとは、独自に構成された反射マーカーのクラスターです。その構成によって、3D 点群内でマーカーの特定とトラッキングが可能となります。複数のリジッドボディを一度に 6 つのパラメータ (位置と向き) でトラッキングすることができます。

以下のセクションでは、リジッドボディの定義に関する具体的な事例を紹介し、Rigid Body ソフトウェアを使ってどのようにそれらをトラッキングするかを説明します。

### 6.2.1 リジッドボディを作成する

最高のトラッキングパフォーマンスを得るために、物理的なリジッドボディを作成する際に考慮すべきいくつかの項目を以下にリストします。構成は、トラッキングボリューム、カメラの配置、マーカーを取り付ける物理的な物体の大きさや形、トラッキングボリュームに含まれるその他の物体にも影響を受けます。

#### マーカータイプ

通常、球形の反射マーカーを使用すると、最も安定した正確な 3D トラッキングデータが得られます。半球状のマーカーや平坦なマーカーは、回転時にカメラに映る形が変化する点で望ましくなく、重心と位置を計算する際にも誤差が生じる場合もあります。

小さいリジッドボディの場合は、小さなマーカーを使って、1つのマーカーが他のマーカーを隠してしまう（カメラの視界を遮る）機会を減らすようにします。

カメラから離れてトラッキングする際は、大きなマーカーを使うとトラッキングの解像度を改善することができます。通常、マーカーの大きさは 10 mm から 25 mm、あるいはそれ以上です。

#### マーカーの量

リジッドボディを定義するためには、少なくとも 3 つのマーカーが必要です。マーカーを多く使うことによって、精度を向上させて、リジッドボディの反転の可能性を減らすことができます。また、3 つ以上のマーカーの使用は余分ですが、割り当てたマーカーのいくつかが見えない場合でも、トラッキングが可能になります（トラッキングのために見えていなければならぬ最小数は 3 です）。

#### リジッドボディのサイズ

マーカーの物理的な配置に関しては、間隔が狭すぎないように、マーカーを互いに近づけすぎないようにします。マーカー同士を 6 ミリメートルよりも近くしてしまうと、結果として次のような問題が発生します。

- ◆ カメラから見た場合、マーカーが重なったりお互いを隠したりする可能性があり、結果として不正な 3D 位置データが得られたり、マーカーのトラッキングが失敗したりします。
- ◆ 点群トラッキングの [Residual]（残差）の設定値が大きすぎたり、キャリブレーションが良好でない場合は、隣接したマーカーに対する光線が同じものと解釈されてしまい、結果としてマーカーの位置の特定を誤ったり、マーカーが誤って報告されたりすることになります。

#### マーカーの配置

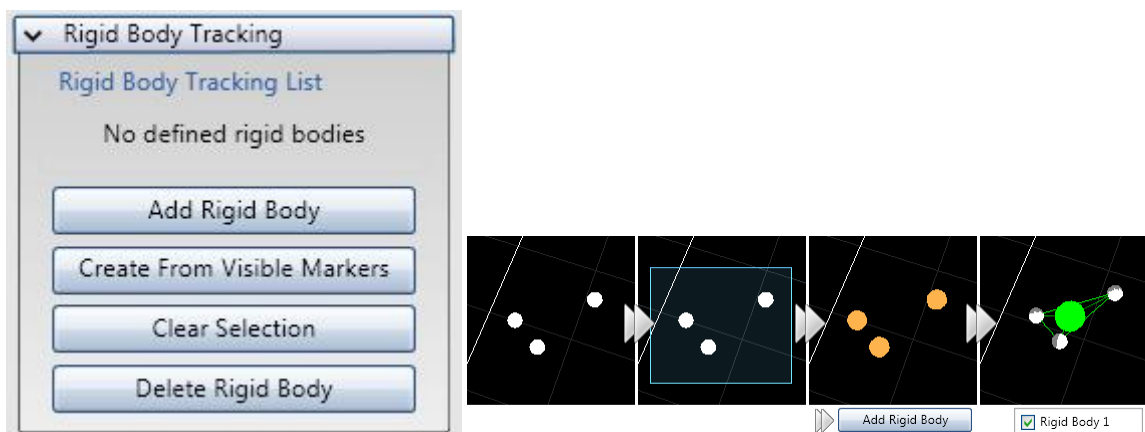
3 つのマーカーを持つリジッドボディについては、マーカーの配置を非対称にすることによって、トラッキングの信頼性が向上します。対象な三角形にすると、ソフトウェアがリジッドボディの正しい向きを特定するのが困難になり、不正な結果に終わる可能性が高くなります。これは、多くの場合、フレーム間でリジッドボディが対称軸に関して反転する現象として現れます。

複数のリジッドボディを同時にトラッキングする場合、できるだけ類似したリジッドボディを作成しないようにしてください。マーカーの配置とサイズを個別に異なるものにするによって、識別の誤りや取り違えの可能性が減少します。

### 6.2.2 リジッドボディの定義と削除

リジッドボディのトラッキングを開始する前に、Point Cloud のデータを使ってソフトウェアでリジッドボディを作成する必要があります。

最初の手順は、あらかじめ保存されている点群データをロードするか、キャリブレーションをロードして新しく点群データのキャプチャを開始することです。作業に使用する点群データが準備できたら、リジッドボディを再定義する最も簡単な方法は、ビューポートでマーカーを選択して、右のパネルの[Add Rigid Body]（リジッドボディを追加）ボタンを押すことです。これで、選択したマーカーの形状の新しいリジッドボディが作成されます。ビューポートでマーカーを選択すると、リアルタイムのキャプチャまたは再生は、自動的に一時停止します。



[Shift] キーを押したままマウス左ボタンでマーカーを囲むように四角形を描くと、複数のマーカーを選択できます。[Ctrl] キーを押したままマウス左ボタンでマーカーをクリックすると、個別のマーカーを選択／選択解除できます。マーカーを選択すると、そのマーカーは色が変わります。

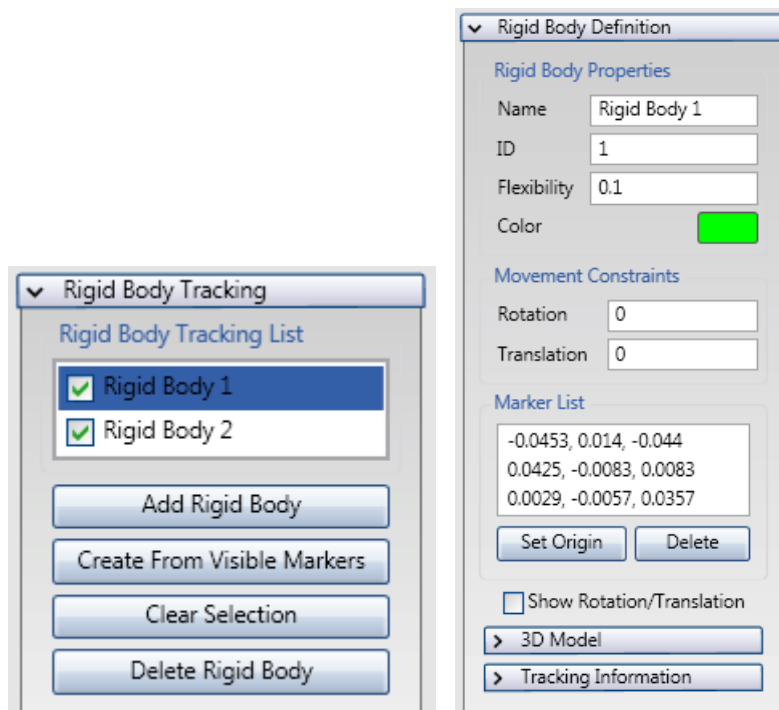
[Create From Visible Markers]（可視のマーカーから作成）ボタンをクリックすると、3Dビューポートに表示されているすべてのマーカーを使ってリジッドボディを作成できます。

新しいリジッドボディをプレビューするには、記録済みデータまたはライブデータの再生を再開します。それによってリジッドボディのトラッキングが開始されます。リジッドボディが作成されると、右のパネルにリストで表示されます。

リジッドボディの名前をクリックして[Delete Rigid Body]（リジッドボディを削除）ボタンを押すことによって、リジッドボディを削除することができます。

### 6.2.3 リジッドボディの編集

作成したリジッドボディは、リジッドボディのリストから名前を選択して変更することができます。



- ◆ **Rigid Body Properties (プロパティ)** : リジッドボディの名前 (Name)、ID、表示色 (Color)、トラッキングの柔軟性 (Flexibility) を変更できます。トラッキングの柔軟性では、どの程度まで歪んだら、リジッドボディのトラッキングを停止するかを制御します。許容値を大きく設定することで、物理的な形状が歪められた場合にもリジッドボディを良好にトラッキングすることができますが、その反面、誤ったマーカーがリジッドボディだとしてトラッキングされたり、誤った向きに解釈されたりする場合があります。柔軟性の値の有効な範囲は 0 から 1.0 で、1.0 がより柔軟です。

- ◆ **Movement Constraints (動きの制約)** : リジッドボディがどの程度移動 (X, Y, Z での移動) し、回転 (ヘディング、アティチュード、バンク) するかに関する制約を設定できます。これによって、リジッドボディトラッキングに対して、物理的なリジッドボディがどの程度の速度で移動し、回転するかの制限をソフトウェア上で課すことができます。これらの制約を調整することで、トラッキングがより安定します。

回転の制約の値は、フレームごとに 0 から 180 度の範囲の角度で表されます。値を 0 にすると、回転の制約が無効になります。移動の制約の値は、フレームごとにメートル単位で設定し、値を 0 にすると制約が無効になります。

- ◆ **Marker List (マーカーリスト)** : このリストは、リジッドボディで使うすべてのマーカーと、その位置を保持します。リストのマーカーを選択して [Set Origin] (原点の設定) ボタンを押すと、リジッドボディの原点を変更できます。また、リストのマーカーを選択して [Delete] (削除) ボタンを押すと、マーカーをリジッドボディから削除できます。

- ◆ **3D Model (3D モデル)** : カスタム 3D モデルをロードしてリジッドボディに割り当てると、より直感的なフィードバックを得ることができます。リジッドボディに割り当てられたモデルは、関連付けられたリジッドボディの動きに基づいて、3D ビューポート内で移動および回転します。

3D モデルの移動はメートル単位、3D モデルの回転は-180 から+180 の範囲で、スライダで調整します。

#### 6.2.4 リジッドボディのロードと保存

ソフトウェアで作成したリジッドボディの定義は、ファイルに保存して後で利用したり、手で編集でき、ソフトウェアに再度ロードし直すことも簡単に行えます。

ソフトウェアで現在定義しているすべてのリジッドボディを保存するためには、以下の手順に従ってください。

[File] (ファイル) メニューから [Save Rigid Body Definitions] (リジッドボディの定義を保存) を選択します。

ファイル名を入力して [Save] (保存) ボタンを押します。

あらかじめ定義してあるリジッドボディをソフトウェアへロードするためには、以下の手順に従います。リジッドボディの定義をロードすると、ソフトウェアで現在定義しているリジッドボディを置き換えてしまうことに注意してください。リジッドボディがロードされると、右側のパネルのリストに表示されます。

[File] (ファイル) メニューから [Load Rigid Body Definitions] (リジッドボディの定義をロード) を選択します。

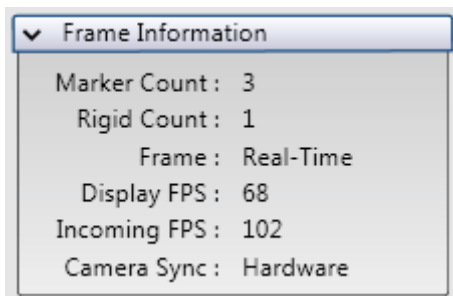
リジッドボディを定義しているファイルを選択して [Open] (開く) ボタンを押します。

### 6.3 トラッキングと記録

OptiTrack Rigid Body ソフトウェアでは、3D 点群データを記録して、オフラインでそれを再生する、リアルタイムトラッキングが可能です。次のセクションでは、データプレビュー、キャプチャ、他のアプリケーションへのエクスポートについて説明します。

#### 6.3.1 トラッキングの状態

カメラ、3D 点群の再構築、再生に関する状態は、[Frame Information] (フレーム情報) 領域に表示されます。ここで示されるフィードバックは、システムの状態とトラブルシューティングに役立ちます。



- ◆ **Marker Count (マーカー数)** : 3D 点群の現在のフレームに含まれるマーカーの数を表します。
- ◆ **Rigid Count (リジッド数)** : 3D 点群の現在のフレームに含まれるリジッドボディの数を表します。ソフトウェアで定義されているけれども、現在のフレームでは見つからないリジッドボディは、合計数に含まれません。
- ◆ **Frame (フレーム)** : 記録データの再生においては、現在のフレーム番号が表示されます。リアルタイムモードでカメラを操作する際は、「Real-Time」(リアルタイム)と表示されます。
- ◆ **Display FPS (表示 FPS)** : 3D ビューポートが更新される、1 秒あたりのフレームレートを表します。
- ◆ **Incoming FPS (入力 FPS)** : 3D 点群データが処理される、1 秒あたりのフレームレートを表します。V100 カメラを使用する際は、この数が「100」に近い値に設定されています。大量のマーカーとリジッドボディがある場合は、システムの負荷が高まり、この数が減る可能性があります。これは、いくつかのフレームが省略されたり、落とされたりしていることを意味します。
- ◆ **Camera Sync (カメラのシンク)** : カメラのシンク状態を表します。カメラシンクケーブルが接続されていて、カメラがハードウェアシンクモードで動作している場合は、[Hardware] (ハードウェア)と表示されます。ハードウェアシンクモードでカメラを使用すると、最高品質の 3D 点群データを作成することができます。カメラシンクケーブルがすべてのカメラの間に接続されていない場合は、ソフトウェアが各フレームごとにデータの到着時間によってカメラをシンクさせようと試みます。ソフトウェアシンクが実行されている場合は、[Synchronized] (シンク中)と表示されます。

### 6.3.2 リアルタイムトラッキング

カメラからキャプチャされると同時に 3D 点群データを処理および表示することは、リアルタイム操作と呼ばれます。カメラの視野内でマーカーが動くと、それらの現在の位置の 3D 点群が作成されます。現在定義されているリジッドボディの位置を決定するために、その 3D 点群に対して検索が実行されます。このデータは後で使用するために記録することも、また、カメラから入力されると同時に他のアプリケーションへストリーミングすることもできます。

#### カメラキャリブレーションのロードとカメラの起動

ソフトウェアがリアルタイムトラッキングを開始する前に、Point Cloud Calibration ツールを使ってカメラをキャリブレーションする必要があります (Section 4.2「カメラキャリブレーション」)

シオンツール」参照)。キャリブレーションが完了したら、ファイルへ保存します。このファイルを Rigid Body ソフトウェアにロードするには、[Tracking] (トラッキング) メニューから [Load Calibration Profile] (キャリブレーションプロファイルのロード) を選択します。キャリブレーションプロファイルがロードされると、ソフトウェアは、3D 点群の作成とビューポートでの表示を開始します。

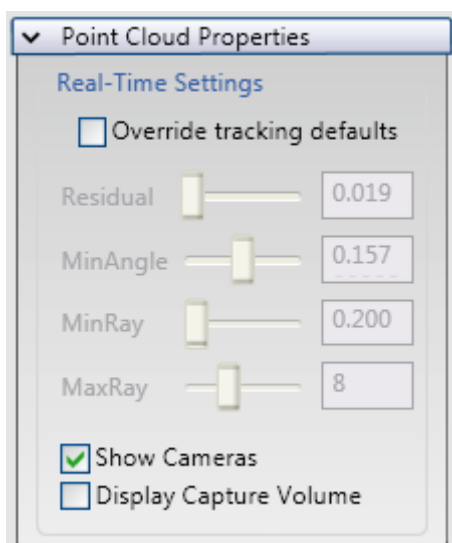
カメラからのトラッキングデータの収集を停止するには、[Tracking] (トラッキング) メニューから [Shutdown Cameras] (カメラのシャットダウン) を選択します。シャットダウンの実行後は、トラッキングを再開するためには、キャリブレーションファイルを再びロードする必要があります。

### 3D 点群

Point Cloud エンジン、2D のマーカーデータを個々のカメラから収集し、識別したマーカーを含む 3D 点群に最構築します。そして、リジッドボディソルバーがこの 3D 点群を検索し、現在定義されているリジッドボディを検出します。

同一の物理的なマーカーを異なるアングルから観察しているカメラは、それらの位置に関する既知の情報を使って、3D 空間でのマーカーの位置を三角測量することができます。カメラの焦点から、カメラの視野内の各 2D マーカーの位置へ引いた仮想的な線を、レイと呼びます。異なるカメラからの複数のレイが所定の距離以内で交差すると、3D 点が形成されます。他のレイと交わらないレイ、あるいは他のレイから遠く離れた場所で交差するレイは、3D マーカーの形成には使われません。

通常、点群のトラッキングの設定はデフォルトのままで十分ですが、特定のカメラに関するパフォーマンスとリジッドボディの構成を調整するためには、変更することも可能です。これらの設定を変更するには、[Point Cloud Properties] (点群プロパティ) タブを開き、[Override Tracking defaults] (トラッキングのデフォルトのオーバーライド) のチェックボックスをオンにします。



- ◆ **Residual (残差)** : 3D 点を形成するためには、カメラからマーカーへのレイがこの値より近くで交差する必要があります (メートル単位)。この値が大きすぎると、誤って検出された 3D マーカーの数が増加します。この値が小さすぎると、特にカメラがうまくキャリブレーションされていない場合に、3D マーカーが形成できません。
- ◆ **Min Angle (最小角度)** : 3D 点を形成するためには、カメラからマーカーへのレイがこの値よりも大きい角度で交差する必要があります (度単位)。同一のマーカーを観察しているカメラが非常に近接して設置されている場合には、この値を小さくする必要あるかもしれません。
- ◆ **Min Ray (最小レイ)** : 3D 点を形成するためには、カメラからマーカーへのレイが、カメラからこの距離より遠くで交差する必要があります (メートル単位)。
- ◆ **Max Ray (最大レイ)** : 3D 点を作成するためには、カメラからマーカーへのレイが、カメラからこの距離よりも近くで交わらなければなりません (メートル単位)。
- ◆ **Show Cameras (カメラの表示)** : ビューポートでカメラ自体の 3D 表示を行うかどうかを制御します。
- ◆ **Display Capture Volume (キャプチャボリュームの表示)** : これによって、キャプチャボリュームの物理的領域を可視化することができます。複数のカメラの視野が重複している 3D 領域を推定することによって、キャプチャボリュームが生成されます。これは、より良好なトラッキング範囲がどの領域にあるかを見つけ出すために使うことができます。

### 6.3.3 記録と再生

カメラからのリアルタイム 3D 点群データは、記録しておいてオフラインで再生することができます。記録したデータを利用する場合は、カメラからのライブデータを使う場合と同様にリジッドボディがトラッキングされます。記録データを利用すると、必要に応じて何度でも、データを逆再生したり、特定のモーションの詳細な動きを見直したりすることができます。記録したデータは他のアプリケーションにストリーミングできるので、リジッドボディトラッキングデータを使用する、ダウンストリーミングツールのテストや検証にも利用することができます。

#### 再生ボタンコントロール



1. 最初のフレームへ巻き戻す
2. 逆再生
3. 停止
4. 一時停止
5. 再生
6. 最後のフレームへ進む
7. 記録

## トラッキングデータの記録方法

3D 点群データを記録する前に、キャリブレーションファイルをロードしてカメラがトラッキングを開始していることを確認します (Section 6.3.2「リアルタイムトラッキング」参照)。

記録を開始するには、記録ボタンを押します。必要量のデータをキャプチャしたら、[Record] (記録) ボタンまたは [Stop] (停止) ボタンを押して、記録を停止することができます。記録されたデータを保存するには、[File] (ファイル) メニューから [Save Capture Data] (キャプチャデータの保存) を選択し、ファイル名を入力して [保存] ボタンを押します。

## 記録データの再生

記録した 3D 点群データがすでにロードされている場合は、[Play] (再生) ボタンを押すことによって再生を開始できます。データを逆方向に再生するためには、[Play Backward] (逆再生) ボタンを押します。データがまだロードされていない場合は、[File] (ファイル) メニューから [Load Capture Data] (キャプチャデータのロード) を選択し、ファイルを選択して [Open] (開く) ボタンを押します。

データの再生中は、3D ビューポートに映像が表示され、タイムラインスクラバーが前方に進みます。[Frame Information] (フレーム情報) タブには、現在のフレーム番号が表示されず (タブが拡張されている場合)。[Pause] (一時停止) ボタンまたは [Stop] (停止) ボタンで、再生を停止することができます。

## リアルタイム再生と記録データ再生の切り替え

Rigid Body ソフトウェアでは、リアルタイムデータのキャプチャと記録データの再生を簡単に切り替えることができます。ソフトウェアがリアルタイムモードの場合は [Real-time] と表示され、それ以外は、[Frame] + 現在のフレーム番号が表示されます。

ソフトウェアに記録データがロードされている状態であれば、どちらかの再生ボタンを押すことによって再生できます。[Pause] (一時停止) ボタンでは再生を一時的に停止することはできますが、リアルタイムモードへは切り替わりません。[Stop] (停止) ボタンを押すと、再び他のボタンが押されるまで、ソフトウェアはリアルタイムモードに切り替わります。

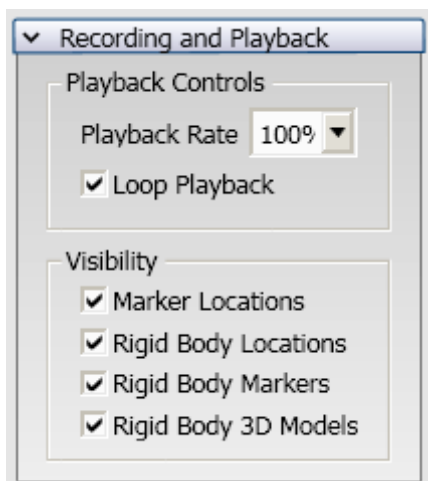
## 記録データの編集

Rigid Body ソフトウェアでは、記録データを編集するための基本的なツールを使用することができます。これらのツールは、[Edit] (編集) メニューの [Recorded Data] (記録データ) セクションの下にあります。

- ◆ **Delete All Frames (すべてのフレームを削除)** : 記録したフレームデータをすべて削除します。
- ◆ **Delete Before Current Frame (現在のフレームより前を削除)** : 現在表示されているフレームより前のフレームデータの記録をすべて削除します。
- ◆ **Delete After Current Frame (現在のフレーム以降を削除)** : 現在表示されているフレームより後のフレームデータの記録をすべて削除します。

## 記録と再生のオプション

記録と再生に関しては、[Recording and Playback]（記録と再生）タブに一連のオプションがあります。



- ◆ **Playback Rate (再生レート)** : 記録済みの 3D 点群データの再生速度は調整することができます。値を小さくするとデータの再生が遅くなり、大きくすると速く再生されます。50%にすると速度は実際の半分、200%では 2 倍になります。
- ◆ **Loop Playback (ループ再生)** : この設定にチェックが入っている場合は、記録データの最後のフレームに到達すると、最初のフレームに戻って再生が続けられます。
- ◆ **Marker Locations (マーカーの位置)** : 3D 点群マーカーの場所を 3D ビューポートに表示するかどうかを制御します。
- ◆ **Rigid Body Locations (リジッドボディの位置)** : 識別されたリジッドボディを 3D ビューポートに表示するかどうかを制御します。
- ◆ **Rigid Body Markers (リジッドボディのマーカー)** : リジッドボディを構成しているマーカーの位置を 3D ビューポートに表示するかどうかを制御します。この表示は、物理的なリジッドボディのマーカーの位置が、期待したマーカーの位置とどの程度異なるかを判断するのに役立ちます。
- ◆ **Rigid Body 3D Models (リジッドボディ 3D モデル)** : リジッドボディに割り当てた 3D モデルを、3D ビューポートに表示するかどうかを制御します。

#### 6.3.4 トラッキングデータのエクスポートとストリーミング

3D 点群とリジッドボディのトラッキングデータは、他のアプリケーションで使用するために、ファイルにエクスポートしたり、リアルタイムでストリーミングすることができます。これらの機能の詳細な使用方法を以下に説明します。

##### CSV フォーマットへのエクスポート

リジッドボディの位置を含む、記録済みのトラッキングデータは、他のアプリケーションで使用するために CSV フォーマットでエクスポートすることができます。エクスポートされたファイルの冒頭には、データフォーマットを記述するコメントが付けられます。

データを CSV ファイルへエクスポートする前に、記録済みデータがロードされており、リジッドボディが定義されていることを確認します。次に、[File] (ファイル) メニューから [Export] (エクスポート) を選択し、ファイル名を入力して [Save] (保存) ボタンを押します。

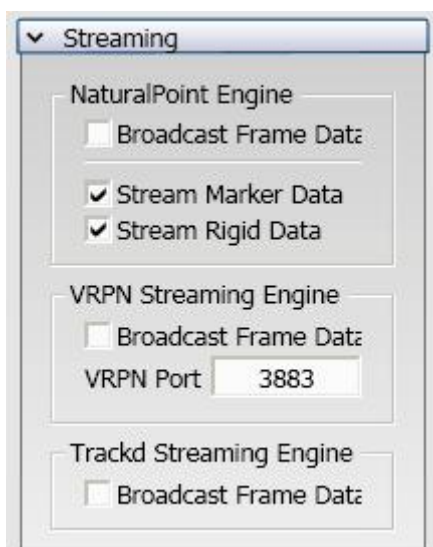
### トラッキングデータのストリーミング

リアルタイムデータと記録済みデータは、他のアプリケーションで使用するためにネットワークを介してストリーミングすることができます。業界標準の VRPN と Trackd を含め、いくつかのストリーミング方法がサポートされています。すべてのストリーミング方法はネットワークトランスポート上に構築されているため、ローカルコンピュータと同様にリモートネットワークコンピュータでもデータを利用することができます。

ストリーミングを試す前に、記録済みデータがロードされているか、カメラがリアルタイムでトラッキングを行っていることを確認します。また、リジッドボディが定義されている必要もあります。

データは TCP/IP を使用してネットワーク上にストリーミングされるので、ファイヤーウォールの設定を見直す必要がある場合があります。ファイヤーウォールがトラフィックをブロックしている場合は、Rigid Body ソフトウェアが適切にデータを送信できなかったり、クライアントアプリケーションがそれを読めなかったりする可能性があります。

ストリーミングを制御する設定は、[Streaming] (ストリーミング) タブにあります。



- ◆ **NaturalPoint Engine (NaturalPoint エンジン)** : NaturalPoint ストリーミングエンジンは、カスタムのストリーミングの実装で、リアルタイムデータおよび記録済みのトラッキングデータに、ネットワークを介してアクセスするために使用します。NaturalPoint は、ストリーミングデータを受け取るクライアントを記述するためのサンプルコードを提供しています。ストリーミングが適切に機能していることを検証するために、標準のサンプルクライアントを使用できます。

[Broadcast Frame Data] (フレームデータのブロードキャスト) チェックボックスをオンにすると、ストリーミングが有効になります。

- ◆ **VRPN エンジン** : VRPN はオープンソースのクラスとプロトコルで、リアルタイムデータおよび記録済みのトラッキングデータにネットワークを介してアクセスするために使用できます。VRPN は低レイテンシを特徴としており、一時的に接続が切れている場合の自動再ネゴシエーション機能が組み込まれています。NaturalPoint では、トラッキングデータを受け取るリスナーのサンプルコードを提供しています。その他のサンプルコードは VRPN の Web サイトで入手可能です。

VRPN では、オブジェクトの識別とアクセスは名前によって行われます。Rigid Body ソフトウェアでの VRPN の実装では、[Rigid Body Definition] (リジッドボディの定義) タブで、リジッドボディに割り当てた名前を使用します。

[Broadcast Frame Data] (フレームデータのブロードキャスト) チェックボックスをオンにすると、ストリーミングが有効になります。

必要に応じて、ポート番号設定を編集すると、VRPN ポートをデフォルト以外の値に変更することができます。

- ◆ **Trackd Engine (Trackd エンジン)** : Trackd は VRCO/Mechdyne によって作成された標準であり、リアルタイムデータおよび記録済みのトラッキングデータにネットワークを介してアクセスするために使用することができます。NaturalPoint では、トラッキングデータを受け取るリスナーのサンプルコードを提供しています。

データをストリーミングする前に、以下の手順に従って Trackd サーバを設定する必要があります。

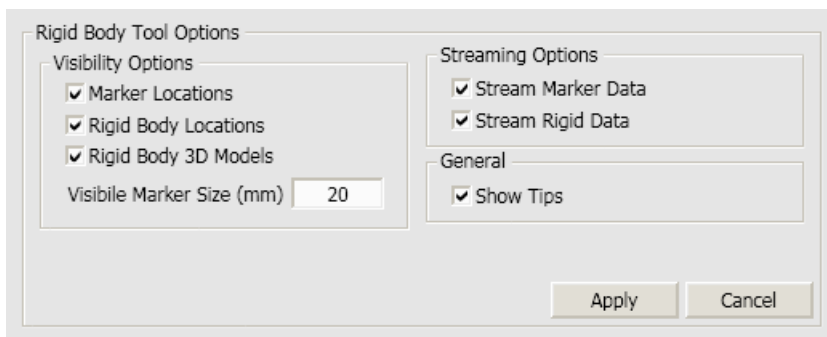
1. Trackd モジュール (dll) とサンプル設定ファイルは、NaturalPoint によって提供されています。
2. 以降の設定では、Trackd を「naturalpointtracker」と呼ぶことにします。
3. トラッキングしたいリジッドボディの数を定義します。
4. ストリーミングのホストはデフォルトでは localhost になっていますが、異なるネットワークアドレスに変更することができます。
5. 設定が完了したら、Trackd サーバを実行します。

[Broadcast Frame Data] (フレームデータのブロードキャスト) チェックボックスをオンにすると、ストリーミングが有効になります。

#### 6.4 アプリケーションのプリファレンス

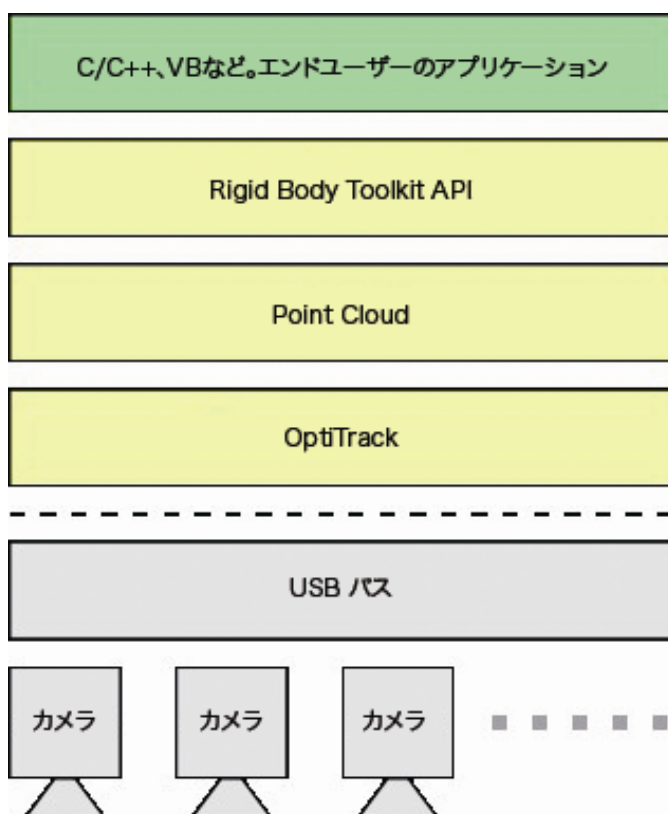
使用目的に応じて、Rigid Body ソフトウェアの設定をカスタマイズすることができます。カスタマイズの制御は、[Edit] (編集) メニューから [Preferences] (プリファレンス) を選択

して表示されるウィンドウで設定できます。これらの設定への変更は、ソフトウェアの終了時に保持され、再開すると復元されます。



## 7. Rigid Body API の使用

お使いのアプリケーションでリジッドボディトラッキングを使用するためには、Rigid Body API を使って、それに対するサポートを実装する必要があります。Rigid Body API は、C/C++のファンクションコールとローダブル DLL モジュールとして記述されています。次のセクションでは、システムアーキテクチャの概要と API の呼び出しに関する仕様の詳細を説明します。



## 7.1 準備作業

ほんの少量のコードを書けば、リジッドボディのトラッキングを開始できます。以下に、初期化の手続きの概要を示します。

ファイルに保存された結果を使ってカメラがキャリブレーションされていることを確認します (Section 4.2 「カメラキャリブレーションツール」参照)。

`RB_InitializeRigidBody()` で、リジッドボディトラッキングを初期化します。

`RB_LoadProfile()` で、既存のキャリブレーションプロファイルをロードします。

`RB_LoadDefinition()` で、現存するリジッドボディ定義をロードします。

`RB_StartCameras()` で、カメラを起動します。

この時点で、カメラは初期化されてフレーム情報を収集しているはずですが、アプリケーションのメインループでは、`RB_GetNextFrame()` または `RB_GetLatestFrame()` を使ってフレームをポーリングする必要があります。

データ処理が完了したら、`RB_StopCameras()` を呼び出し、次に `RB_ShutdownRigidBody` を呼び出します。

## 7.2 Rigid Body API の呼び出し

### 7.2.1 Rigid Body の開始と終了

#### 7.2.1.1 `RB_InitializeRigidBody()`

`RB_InitializeRigidBody()` 関数は、Rigid Body トラッキング API の初期化を試行します。これは、API の他のコンポーネントを使う前に呼び出す必要があります。この関数は、成功したかどうかの情報を返します。

```
NPRESULT RB_InitializeRigidBody()
```

**パラメータ**  
・ なし

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功

#### 7.2.1.2 RB\_ShutdownRigidBody()

RB\_ShutdownRigidBody()関数は、Rigid Body トラッキング API の終了を試行します。これは、RB\_StopCameras()の後、かつ、Rigid Body API を使うアプリケーションを終了する前に呼び出す必要があります。この関数は、成功したかどうかの情報を返しません。

NPRESULT RB\_ShutdownRigidBody()

### パラメータ

- なし

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功

#### 7.2.2 Rigid Body インタフェース

##### 7.2.2.1 RB\_LoadProfile()

RB\_LoadProfile()関数は、ファイルに保存されているキャリブレーションプロファイルのロードを試行します。プロファイルへのパスを取り、成功したかどうかの情報を返します。

### パラメータ

- filename  
[in]キャリブレーションプロファイルの場所へのパス

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功
InvalidFile	指定されたファイルやパスが有効ではない
InvalidLicense	有効な Rigid Body ライセンスが見つからない
InvalidProfileCameraCount	指定されたプロファイルが現在のカメラ設定に適合しない

### 7.2.2.2 RB\_StartCameras()

RB\_StartCameras() 関数は、カメラを起動してトラッキングを開始します。この関数は RB\_LoadProfile() の後に呼び出す必要があります。この関数は、成功したかどうかの情報を返します。

NPRESULT RB\_StartCameras()

### パラメータ

- なし

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功
PointCloudNotInitialized	Point Cloud エンジンが適切に初期化されていない
CamerasAlreadyStarted	カメラはすでに起動済み
NoProfileLoaded	カメラに対するキャリブレーションプロファイルがロードされていない
UnableToInitializeOptiTrackCameras	カメラの初期化の際に問題が発生した

### 7.2.2.3 RB\_StopCameras()

RB\_StopCameras() 関数は、カメラを停止してトラッキングを停止します。この関数は、成功したかどうかの情報を返します。

NPRESULT RB\_StopCameras()

### パラメータ

- なし

### パラメータ

値	意味
<b>NPRESULT_SUCCESS</b>	メソッドは成功
<b>PointCloudNotInitialized</b>	Point Cloud エンジンが適切に初期化されていない
<b>CamerasNotRunning</b>	カメラがすでに停止している

#### 7.2.2.4 RB\_GetNextFrame()

RB\_GetNextFrame() 関数は、次に使用可能なトラッキングデータのフレームをロードします。フレームに関する情報には、リジッドボディ制御の API 呼び出しを使用してアクセスします。この関数は、成功したかどうかの情報を返します。

**NPRESULT RB\_GetNextFrame()**

### パラメータ

- なし

### パラメータ

値	意味
<b>NPRESULT_SUCCESS</b>	メソッドは成功
<b>InvalidLicense</b>	有効な Rigid Body ライセンスが見つからない
<b>NoFrameAvailable</b>	トラッキングデータが見つからない

#### 7.2.2.5 RB\_GetLatestFrame()

RB\_GetLatestFrame() 関数は、最新のトラッキングデータのフレームをロードします。フレームに関する情報には、リジッドボディ制御の API 呼び出しを使用してアクセスします。この関数は、成功したかどうかの情報を返します。

**NPRESULT RB\_GetLatestFrame()**

### パラメータ

- なし

### パラメータ

値	意味
<b>NPRESULT_SUCCESS</b>	メソッドは成功
<b>InvalidLicense</b>	有効な Rigid Body ライセンスが見つからない
<b>NoFrameAvailable</b>	トラッキングデータが見つからない

#### 7.2.2.6 RB\_LoadDefinition()

RB\_LoadDefinition() 関数は、トラッキングのためのリジッドボディ定義をファイルからロードします。この関数は、成功したかどうかの情報を返します。

```
NPRESULT RB_LoadDefinition(const char *filename)
```

### パラメータ

- filename  
[in] リジッドボディ定義ファイルの場所へのパス

### パラメータ

値	意味
<b>NPRESULT_SUCCESS</b>	メソッドは成功
<b>Failed</b>	定義のロードに失敗

#### 7.2.2.7 RB\_SaveDefinition()

RB\_SaveDefinition() 関数は、現在ロードされているリジッドボディ定義を、後で利用するためにファイルに保存します。この関数は、成功したかどうかの情報を返します。

```
NPRESULT RB_SaveDefinition(const char *filename)
```

### パラメータ

- filename  
[in] リジッドボディ定義ファイルの場所へのパスです。

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功
Failed	定義の保存に失敗

## 7.2.3 リジッドボディストリーミング

### 7.2.3.1 RB\_StreamTrackd()

RB\_StreamTrackd()関数は、Trackd インタフェースを介したトラッキングデータのストリーミングを開始および停止します。この関数は、成功したかどうかの情報を返します。

```
NPRESULT RB_StreamTrackd(bool enabled)
```

#### パラメータ

- ・ enabled  
[in]ストリーミングを開始/停止します。true で開始し、false で停止します。

#### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功

### 7.2.3.2 RB\_StreamVRPN()

RB\_StreamVRPN()関数は、VRPN インタフェースを介したトラッキングデータのストリーミングを開始および停止します。この関数は、成功したかどうかの情報を返します。

```
NPRESULT RB_StreamVRPN(bool enabled, int port)
```

#### パラメータ

- ・ enabled  
[in]ストリーミングを開始/停止します。true で開始、false で停止。
- ・ port  
[in]ブロードキャストに使用するネットワークポートを選択します。

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功

#### 7.2.3.3 RB\_StreamNP()

RB\_StreamNP()関数は、NaturalPoint ストリーミングインタフェースを介したトラッキングデータのストリーミングを開始および停止します。この関数は、成功したかどうかの情報を返します。

```
NPRESULT RB_StreamNP(bool enabled)
```

### パラメータ

- enabled  
[in]ストリーミングを開始/停止します。true で開始、false で停止。

### パラメータ

値	意味
NPRESULT_SUCCESS	メソッドは成功

#### 7.2.4 リジッドボディフレーム

##### 7.2.4.1 RB\_FrameMarkerCount()

RB\_FrameMarkerCount()関数は、現在のフレームで見つかった 3D 点群マーカーの数を返します。

```
int RB_FrameMarkerCount()
```

### パラメータ

- (戻り値)  
[out]現在のフレームで見つかった 3D 点群マーカーの数です。

##### 7.2.4.2 RB\_FrameMarkerX()

RB\_FrameMarkerX()関数は、選択された 3D 点群マーカーの X 軸上の位置をメートル単位で返します。

```
float RB_FrameMarkerX(int index)
```

### パラメータ

- (戻り値)  
[out]選択された 3D 点群マーカーの X 軸上での位置です (メートル単位)。

- ・ index  
[in]取得するマーカーのインデックス番号です（インデックスは0が基準）。

#### パラメータ

値	意味
0	フレームデータが見つからない

#### 7.2.4.3 RB\_FrameMarkerY()

RB\_FrameMarkerY()関数は、選択された3D点群マーカーのY軸上の位置をメートル単位で返します。

```
float RB_FrameMarkerY(int index)
```

#### パラメータ

- ・ (戻り値)  
[out]選択された3D点群マーカーのY軸上での位置です（メートル単位）。
- ・ index  
[in]取得するマーカーのインデックス番号を返します（インデックスは0が基準）。

#### パラメータ

値	意味
0	フレームデータが見つからない

#### 7.2.4.4 RB\_FrameMarkerZ()

RB\_FrameMarkerZ()関数は、選択された3D点群マーカーのZ軸上の位置をメートル単位で返します。

```
float RB_FrameMarkerZ(int index)
```

#### パラメータ

- ・ (戻り値)  
[out]選択された3D点群マーカーのZ軸上での位置です（メートル単位）。
- ・ index  
[in]取得するマーカーのインデックス番号を返します（インデックスは0が基準）。

### パラメータ

値	意味
0	フレームデータが見つからない

## 7.2.5 リジッドボディ制御

### 7.2.5.1 RB\_IsRigidBodyTracked()

RB\_IsRigidBodyTracked() 関数は、選択されたリジッドボディが現在のフレームにあるかどうかの情報を返します。

```
bool RB_IsRigidBodyTracked(int index)
```

### パラメータ

- ・ (戻り値)  
[out] 選択したリジッドボディが現在のフレームに見つければ true を返します。
- ・ index  
[in] 取得するリジッドボディのインデックス番号です (インデックスは 0 が基準)。

### 7.2.5.2 RB\_GetRigidBodyLocation()

RB\_GetRigidBodyLocation() 関数は、選択したリジッドボディの位置と方向を返します。RB\_IsRigidBodyTracked() は、現在のフレームでリジッドボディがトラッキングされているかどうかを判断するために最初にチェックする必要があります。これを行わないとデータが最新でなくなる可能性があります。

```
void RB_GetRigidBodyLocation(int RigidBodyIndex,  
float *x, float *y, float *z,  
float *qx, float *qy, float *qz, float *qw,  
float *heading, float *attitude, float *bank)
```

### パラメータ

- ・ RigidBodyIndex  
[in] 取得するリジッドボディのインデックス番号です (インデックスは 0 が基準)。
- ・ x  
[out] 選択したリジッドボディの X 軸上の位置を受け取ります (メートル単位)。
- ・ y  
[out] 選択したリジッドボディの Y 軸上の位置を受け取ります (メートル単位)。
- ・ z  
[out] 選択したリジッドボディの Z 軸上の位置を受け取ります (メートル単位)。

- ・ qx  
[out] 選択したリジッドボディのクォータニオンの x 値を受け取ります。
- ・ qy  
[out] 選択したリジッドボディのクォータニオンの y 値を受け取ります。
- ・ qz  
[out] 選択したリジッドボディのクォータニオンの z 値を受け取ります。
- ・ qw  
[out] 選択したリジッドボディのクォータニオンの w 値を受け取ります。
- ・ heading  
[out] 選択したリジッドボディのヘディングの方向の値を受け取ります（度単位）。
- ・ attitude  
[out] 選択したリジッドボディのアティチュードの方向の値を受け取ります（度単位）。
- ・ bank  
[out] 選択したリジッドボディのバンクの方向の値を受け取ります（度単位）。

#### 7.2.5.3 RB\_ClearRigidBodyList()

RB\_ClearRigidBodyList() 関数は、現在ロードされているリジッドボディ定義をすべて削除します。

```
void RB_ClearRigidBodyList()
```

##### パラメータ

- ・ なし

#### 7.2.5.4 RB\_GetRigidBodyCount()

RB\_GetRigidBodyCount() 関数は、現在ロードされているリジッドボディ定義の数を返します。

```
int RB_GetRigidBodyCount()
```

##### パラメータ

- ・ (戻り値)  
[out] 現在ロードされているリジッドボディ定義の数です。

#### 7.2.5.5 RB\_DeleteRigidBodyMarker()

RB\_DeleteRigidBodyMarker() 関数は、リジッドボディ定義からマーカーを削除します。

```
void RB_DeleteRigidBodyMarker(int RigidBodyIndex, int MarkerIndex)
```

#### パラメータ

- ・ RigidBodyIndex  
[in] マーカーを削除する対象のリジッドボディのインデックス番号です（インデックスは0が基準）。
- ・ MarkerIndex  
[in] 削除対象のマーカーのインデックス番号です（インデックスは0が基準）。

#### 7.2.5.6 RB\_AddRigidBodyMarker()

RB\_AddRigidBodyMarker() 関数は、既存のリジッドボディ定義にマーカーを追加します。

```
void RB_AddRigidBodyMarker(int RigidBodyIndex, float x, float y, float z)
```

#### パラメータ

- ・ RigidBodyIndex  
[in] 対象のリジッドボディのインデックス番号です（インデックスは0が基準）。
- ・ x  
[in] 原点を基準とした、マーカーの X 軸上の位置です（メートル単位）。
- ・ y  
[in] 原点を基準とした、マーカーの Y 軸上の位置です（メートル単位）。
- ・ z  
[in] 原点を基準とした、マーカーの Z 軸上の位置です（メートル単位）。

#### 7.2.5.7 RB\_SetRigidBodyOrigin()

RB\_SetRigidBodyOrigin() 関数は、既存のリジッドボディ定義の原点を変更します。

```
void RB_SetRigidBodyOrigin(int RigidBodyIndex, float x, float y, float z)
```

#### パラメータ

- ・ RigidBodyIndex  
[in] 対象のリジッドボディのインデックス番号です（インデックスは0が基準）。
- ・ x  
[in] 新しい原点の X 軸上の位置です（メートル単位）。
- ・ y  
[in] 新しい原点の Y 軸上の位置です（メートル単位）。
- ・ z  
[in] 新しい原点の Z 軸上の位置です（メートル単位）。

#### 7.2.5.8 RB\_GetRigidBodyID()

RB\_GetRigidBodyID() 関数は、選択したリジッドボディの ID を返します。

```
int RB_GetRigidBodyID(int index)
```

##### パラメータ

- ・ (戻り値)  
[out] 選択したリジッドボディの ID です。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。

#### 7.2.5.9 RB\_SetRigidBodyID()

RB\_SetRigidBodyID() 関数は、選択したリジッドボディの ID を変更します。

```
void RB_SetRigidBodyID(int index, int ID)
```

##### パラメータ

- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。
- ・ ID  
[in] 選択したリジッドボディの新しい ID の値です。

#### 7.2.5.10 RB\_CreateRigidBody()

RB\_CreateRigidBody() 関数は、新規のリジッドボディ定義を作成します。

```
void RB_CreateRigidBody(const char *name)
```

##### パラメータ

- ・ name  
[in] 新規に作成したリジッドボディに割り当てる名前です。

#### 7.2.5.11 RB\_DeleteRigidBody()

RB\_DeleteRigidBody() 関数は、既存のリジッドボディ定義を削除します。

```
void RB_DeleteRigidBody(int index)
```

### パラメータ

- ・ index  
[in] 削除対象のリジッドボディのインデックス番号です（インデックスは0が基準）。

#### 7.2.5.12 RB\_GetRigidBodyFlexibility()

RB\_GetRigidBodyID() 関数は、選択したリジッドボディの Flexibility (柔軟性) 設定値を返します。

```
float RB_GetRigidBodyFlexibility(int index)
```

### パラメータ

- ・ (戻り値)  
[out] 選択したリジッドボディの Flexibility の設定値です。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です（インデックスは0が基準）。

#### 7.2.5.13 RB\_SetRigidBodyFlexibility()

RB\_SetRigidBodyFlexibility() 関数は、選択したリジッドボディの Flexibility (柔軟性) 設定値を変更します。

```
void RB_SetRigidBodyFlexibility(int index, float value)
```

### パラメータ

- ・ index  
[in] 対象のリジッドボディのインデックス番号です（インデックスは0が基準）。
- ・ value  
[in] 選択したリジッドボディの新しい Flexibility (柔軟性) 設定です。入力値は0（最も柔軟性が低い）から1.0（最も高い）です。

#### 7.2.5.14 RB\_GetRigidBodyRotationConst()

RB\_GetRigidBodyRotationConst() 関数は、選択したリジッドボディの回転の制約設定を返します。

```
float RB_GetRigidBodyRotationConst(int index)
```

### パラメータ

- ・ (戻り値)  
[out] 選択したリジッドボディの回転の制約設定です。

- ・ index  
[in]対象のリジッドボディのインデックス番号です(インデックスは0が基準)。

#### 7.2.5.15 RB\_SetRigidBodyRotationConst()

RB\_SetRigidBodyRotationConst()関数は、選択したリジッドボディの回転の制約設定を変更します。

```
void RB_SetRigidBodyRotationConst(int index, float value)
```

##### パラメータ

- ・ index  
[in]対象のリジッドボディのインデックス番号です(インデックスは0が基準)。
- ・ value  
[in]選択したリジッドボディの新しい回転制約設定です。フレームごとの角度を0度から180度の間で設定します。値0は制約なし。

#### 7.2.5.16 RB\_GetRigidBodyTranslationConst()

RB\_GetRigidBodyTranslationConst()関数は、選択したリジッドボディの移動の制約設定を返します。

```
float RB_GetRigidBodyTranslationConst(int index)
```

##### パラメータ

- ・ (戻り値)  
[出力]選択したリジッドボディの移動の制約設定です。
- ・ index  
[in]対象のリジッドボディのインデックス番号です(インデックスは0が基準)。

#### 7.2.5.17 RB\_SetRigidBodyTranslationConst()

RB\_SetRigidBodyTranslationConst()関数は、選択したリジッドボディの移動の制約設定を変更します。

```
void RB_SetRigidBodyTranslationConst(int index, float value)
```

##### パラメータ

- ・ index  
[in]対象のリジッドボディのインデックス番号です(インデックスは0が基準)。

- ・ value  
[in] 選択したリジッドボディの新しい移動の制約設定。フレームあたりの値をメートル単位で指定します。値 0 は制約なし。

#### 7.2.5.18 RB\_SetRigidBodyEnabled()

RB\_SetRigidBodyEnabled() 関数は、選択したリジッドボディのトラッキングが可能かどうかを制御します。

```
void RB_SetRigidBodyEnabled(int index, bool enabled)
```

##### パラメータ

- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。
- ・ enabled  
[in] true はリジッドボディのトラッキングを有効にします。false は無効。

#### 7.2.5.19 RB\_GetRigidBodyEnabled()

RB\_GetRigidBodyEnabled() 関数は、選択したリジッドボディがトラッキング可能かどうかの情報を返します。

```
bool RB_GetRigidBodyEnabled(int index)
```

##### パラメータ

- ・ (戻り値)  
[out] true の場合には、リジッドボディのトラッキングが可能です。false の場合は無効。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。

#### 7.2.5.20 RB\_SetRigidBodyColor()

RB\_SetRigidBodyColor() 関数は、選択したリジッドボディを 3D ビューポートに表示する際の色を変更します。

```
void RB_SetRigidBodyColor(int index, int r, int g, int b)
```

##### パラメータ

- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。

- ・ r  
[in] 赤色の成分の値です (0 から 255 の範囲で設定)。
- ・ g  
[in] 緑色の成分の値です (0 から 255 の範囲で設定)。
- ・ b  
[in] 青色の成分の値です (0 から 255 の範囲で設定)。

#### 7.2.5.21 RB\_GetRigidBodyColorR()

RB\_GetRigidBodyColorR() 関数は、選択したリジッドボディを 3D ビューポートに表示する際に使用する色の赤色の成分を返します。

```
int RB_GetRigidBodyColorR(int index)
```

##### パラメータ

- ・ (戻り値)  
[out] リジッドボディの表示色の赤色成分です。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。

#### 7.2.5.22 RB\_GetRigidBodyColorG()

RB\_GetRigidBodyColorG() 関数は、選択したリジッドボディを 3D ビューポートに表示する際に使用する色の緑色の成分を返します。

```
int RB_GetRigidBodyColorG(int index)
```

##### パラメータ

- ・ (戻り値)  
[out] リジッドボディの表示色の緑色成分です。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です (インデックスは 0 が基準)。

#### 7.2.5.23 RB\_GetRigidBodyColorB()

RB\_GetRigidBodyColorG() 関数は、選択したリジッドボディを 3D ビューポートに表示する際に使用する色の青色の成分を返します。

```
int RB_GetRigidBodyColorB(int index)
```

### パラメータ

- ・ (戻り値)  
[out] リジッドボディの表示色の青色成分です。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です(インデックスは0が基準)。

#### 7.2.5.24 RB\_GetRigidBodyMarkerCount()

RB\_GetRigidBodyMarkerCount() 関数は、選択したリジッドボディ定義で使われているマーカーの数を返します。

```
int RB_GetRigidBodyMarkerCount(int index)
```

### パラメータ

- ・ (戻り値)  
[out] 選択したリジッドボディで使われているマーカーの数です。
- ・ index  
[in] 対象のリジッドボディのインデックス番号です(インデックスは0が基準)。

#### 7.2.5.25 RB\_GetRigidBodyMarker()

RB\_GetRigidBodyMarker() 関数は、選択したリジッドボディ定義における、選択したマーカーの位置を返します。マーカーの位置は、リジッドボディの基点または重心からの相対距離の表示です(メートル単位)。

```
void RB_GetRigidBodyMarker(int RigidBodyIndex, int MarkerIndex, float *x, float *y, float *z)
```

### パラメータ

- ・ RigidBodyIndex  
[in] 対象のリジッドボディのインデックス番号です(インデックスは0が基準)。
- ・ MarkerIndex  
[in] 取得するマーカーのインデックス番号です(インデックスは0が基準)。
- ・ x  
[out] マーカーの X 軸上の位置です(メートル単位)。
- ・ y  
[out] マーカーの Y 軸上の位置です(メートル単位)。
- ・ z  
[out] マーカーの Z 軸上の位置です(メートル単位)。

#### 7.2.5.26 RB\_GetSyncQuality()

RB\_GetSyncQuality() 関数は、現在のカメラシンの状態を返します。

```
int RB_GetSyncQuality()
```

##### パラメータ

- ・ (戻り値)  
[out]現在のカメラシンク状態です。0=シンクなし、1=ハードウェアシンク、2=良好なソフトウェアシンク、3=良好でないシンク

#### 7.2.6 Point Cloud インタフェース

##### 7.2.6.1 RB\_SetPointCloudTrackingParams()

RB\_SetPointCloudTrackingParams() 関数は、Point Cloud のトラッキング設定を変更します。

```
void RB_SetPointCloudTrackingParams(int iMinRays, float fMaxResidual, float fMinAngle, float fMinRayLength, float fMaxRayLength)
```

##### パラメータ

- ・ iMinRays  
[in]3D 点群マーカーを形成するために必要なレイの最小数です。
- ・ fMaxResidual  
[in]3D 点を形成するためには、カメラからマーカーへのレイは、この値より近くで交差する必要があります (メートル単位)。
- ・ fMinAngle  
[in]3D 点を形成するためには、カメラからマーカーへのレイがこの値よりも大きい角度で交差する必要があります (度単位)。
- ・ fMinRayLength  
[in]3D 点を形成するためには、カメラからマーカーへのレイが、カメラからこの距離より遠くで交差する必要があります (メートル単位)。
- ・ fMaxRayLength  
[in]3D 点を形成するためには、カメラからマーカーへのレイが、カメラからこの距離より近くで交差する必要があります (メートル単位)。

##### 7.2.6.2 RB\_GetPointCloudTrackingParams()

RB\_GetPointCloudTrackingParams() 関数は、現在の Point Cloud のトラッキング設定を返します。

```
void RB_GetPointCloudTrackingParams(int *iMinRays, float *fMaxResidual, float *fMinAngle, float *fMinRayLength, float *fMaxRayLength)
```

#### パラメータ

- ・ iMinRays  
[out]3D 点群マーカを形成するために必要なレイの最小数です。
- ・ fMaxResidual  
[out]3D 点を形成するためには、カメラからマーカへのレイは、この値より近くで交差する必要があります（メートル単位）。
- ・ fMinAngle  
[out]3D 点を形成するためには、カメラからマーカへのレイがこの値よりも大きい角度で交差する必要があります（度単位）。
- ・ fMinRayLength  
[out]3D 点を形成するためには、カメラからマーカへのレイが、カメラからこの距離より遠くで交差する必要があります（メートル単位）。
- ・ fMaxRayLength  
[out]3D 点を形成するためには、カメラからマーカへのレイが、カメラからこの距離より近くで交差する必要があります（メートル単位）。

#### 7.2.6.3 RB\_CameraCount()

RB\_CameraCount() 関数は、接続されているカメラの数を返します。

```
int RB_CameraCount()
```

#### パラメータ

- ・ (戻り値)  
[out]接続されているカメラの数です。

#### 7.2.6.4 RB\_CameraXLocation()

RB\_CameraXLocation() 関数は、座標系原点からのカメラの X 軸上の位置を返します。

```
float RB_CameraXLocation(int index)
```

#### パラメータ

- ・ (戻り値)  
[out]選択したカメラの X 軸上の位置です（メートル単位）。
- ・ index  
[in]取得するカメラのインデックス番号です（インデックスは 0 が基準）。

#### 7.2.6.5 RB\_CameraYLocation()

RB\_CameraYLocation() 関数は、座標系原点からのカメラの Y 軸上の位置を返します。

```
float RB_CameraYLocation(int index)
```

##### パラメータ

- ・ (戻り値)  
[out] 選択したカメラの Y 軸上の位置です (メートル単位)。
- ・ index  
[in] 取得するカメラのインデックス番号です (インデックスは 0 が基準)。

#### 7.2.6.6 RB\_CameraZLocation()

RB\_CameraZLocation() 関数は、座標系原点からのカメラの Z 軸上の位置を返します。

```
float RB_CameraZLocation(int index)
```

##### パラメータ

- ・ (戻り値)  
[out] 選択したカメラの Z 軸上の位置です (メートル単位)。
- ・ index  
[in] 取得するカメラのインデックス番号です (インデックスは 0 が基準)。

#### 7.2.6.7 RB\_CameraOrientationMatrix()

RB\_CameraOrientationMatrix() 関数は、選択したカメラの方向行列内の、選択した要素を返します。

```
float RB_CameraOrientationMatrix(int camera, int index)
```

##### パラメータ

- ・ (戻り値)  
[out] 方向行列内の選択した要素です。
- ・ camera  
[in] 対象のカメラのインデックス番号です (インデックスは 0 が基準)。
- ・ index  
[in] 取得する配列におけるアイテムのインデックス。配列には 9 つの要素があり、有効な入力の範囲は 0 から 8 です。

## 7.2.7 リターンコード処理

### 7.2.7.1 RB\_GetResultString()

RB\_GetResultString() 関数は、Rigid Body API の他の関数から返されたステータスコードについて、プレーンテキストのメッセージを返します。

```
const char *RB_GetResultString(NPRESULT result)
```

#### パラメータ

- ・ (戻り値)  
[out] 選択したリターンコードのプレーンテキストメッセージです。
- ・ result  
[in] Rigid Body API の他の関数から返されるステータスコードです。

## 8. 使用上のヒント

### 8.1 トラッキング環境

Point Cloud Toolkit と Rigid Body Toolkit は、さまざまな条件下で動作するように設計されています。また、現在の OptiTrack カメラは、以前と比較してより安定し、信頼性も高くなっています。しかし、パフォーマンスを最適化するためにできることは、いくつもあります。

#### カメラとユーザーの間の距離：

OptiTrack カメラとトラッキングマーカの距離の最適な範囲は、0.5 から 6 メートルです。

#### ライティング：

部屋のライトを消すか暗くするかし、OptiTrack カメラの視野に直接入っている反射物を取り除くことをお勧めします。OptiTrack カメラが何を認識しているかは、キャリブレーションツールのカメラプレビューの手順を使用して簡単にチェックすることができます。詳細は Section 4.2.2 を参照してください。

### 8.2 トラッキングマーカ

マーカは、表面がきれいで傷がない状態の場合に、最も良好に検出されます。マーカの反射面が、肌やその他の表面と頻りに接触するような場合は、マーカを定期的に交換することをお勧めします。

## 9. ソフトウェアのアップデート

皆様の要求によりよくお応えするため、Tracking Toolkit ソフトウェアは常にアップデートされています。アップデートは、<http://www.mocap.jp/optitrack/>から無償でダウンロードすることができます。

**新しいソフトウェアをインストールするには、以下の手順に従ってください。**

1. お持ちの製品に対してリストされているソフトウェアのバージョンが、コンピュータにインストールされているソフトウェアのバージョンよりも新しい場合は、アップデートのインストーラファイルをデスクトップまたは一時的なフォルダにダウンロードして保存します。新しいバージョンが提供されていない場合は、ソフトウェアをアップデートする必要はありません。
2. コンピュータから OptiTrack カメラを取り外し、TrackingToolkit ソフトウェアが実行されていないことを確認します。
3. Windows のコントロールパネルの[プログラムの追加と削除]を使用して、インストール済みの Tracking Toolkit ソフトウェアをアンインストールします。
4. 新しいソフトウェアをインストールするために、ダウンロードしたインストーラーをダブルクリックします。プログラムのアイコンがデスクトップに再び表示されたら、コンピュータに OptiTrack カメラを再接続してアップデートされたソフトウェアを起動します。