社団法人 人工知能学会
Japanese Society for
Artificial Intelligence

人工知能学会研究会資料
JSAI Technical Report
SIG-Challenge-B001-6 (5/3)

複数の外部カメラ画像に基づくヒト型ロボットの3次元形状の実時間取得
RoboCupSoccer SSL Humanoidの3次元情報サーバをを目指して

A Realtime Acquisition of 3D Shape of Humanoid Robots Based on Multiple External Camera Images

Aiming at 3D Information Server for the RoboCupSoccer SSL Humanoid

木村 尭海　升谷 保博

Takaumi KIMURA　Yasuhiro MASTANI

大阪電気通信大学

Osaka Electro-Communication University

Abstract

RoboCupSoccer SSL Humanoid, soccer game by humanoid robots using external cameras, was proposed. Although 2D image processing are used in the current phase, in the next phase, the targets are real-time 3D shapes acquisition based on images of multiple cameras and deciding action based on the results. A shared vision system is also the target. However, what to be shared for 3D vision system has not been decided. In this paper, the authors propose that voxel data of 3D space on the game field is shared. Moreover, in order to prove the realization of the proposal, the authors develop a program for cutting voxels whose size is 10[mm] of 3[m]×4[m]×0.5[m] space based on images taken with 4 cameras by visual cone intersection and sending run-length compressed data to clients via network. This program can execute the chain of processing 30times per second on PC with Intel Core i7 950 processor.

1 はじめに

RoboCupSoccer のリーグの一つで，車輪型のロボットがサッカーを行う小型ロボットリーグ（SSL: Small Size robot League）は，他のリーグと異なり，外部カメラを用いてロボットなどの位置を認識して制御を行うおかげで，パスなどの協調的なプレーができるまでに発展してきている．この競技に参加するには，様々なノウハウや経験が必要であるが，その方法は確立しつつあり，研究としては，飽和気味である．また，技術レベルの高度化と必要な費用の増大により新規参入が難しくなっている．

このような現状を踏まえて，SSL の特徴を継承しつつ，新たな研究を展開し，かつ，参加者の裾野を広げることを目的として，SSL Humanoid が提案された[升谷, 2010]．

現状では，SSL と同様に，フィールド上空のカメラでロボットの頭頂部に取り付けられたマーカのパターンを撮影し，2 次元的な画像処理を経て，その番号や位置，方向を算出している．しかし，SSL Humanoid のロードマップでは，2015 年頃には，複数の外部カメラで撮影した画像に基づいて，複数のロボットの 3 次元的な形状を取得し，それに基づきロボットの行動決定を行うことを目指している．

SSL では，2010 年から視覚系の共有が義務付けられる．参加チームは，カメラとその画像を処理する SSL-Vision と呼ばれるサーバプログラムを共有し，SSL-Vision から送られてくるフィールド上の位置情報を利用する[Zickler, 2010]．SSL Humanoid でも視覚系の共有は必須項目である．現状の 2 次元の場合は，SSL-Vision を利用することが決まっているが，3 次元の場合に何をどのように共有するかはまだ決まっていない．

そこで，本稿では，フィールドの 3 次元空間のボクセルデータを共有することを提案する．さらに，それを実現するプログラムを実際に開発し，実現の可能性についても検討する．

以下では，まず，3 次元情報を取得する視覚系を共有する方法について議論し，本稿で提案する方法について述べる．次に，提案の方法を実現するプログラムについて説明し，それを用いた評価実験の結果を示す．最後に，今後の展望や課題について述べる．

2 ボクセルデータサーバの提案

2009 年の SSL Humanoid の試合は，Figure 1 のような形態で行われた．ロボットの頭頂部にはマーカ板が取り付けられている．それをフィールド上空のカメラで撮影して，各オブジェクトのフィールド上の 2 次元的な位置と方向

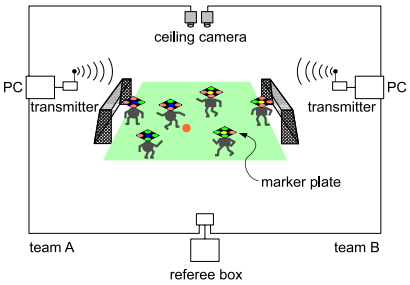

Figure 1: Initial phase of the SSL Humanoid (2009)

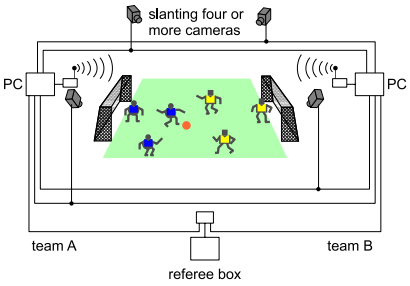

Figure 2: Final phase of the SSL Humanoid (2015)

を得ている．カメラや画像認識のプログラムは各チームが別々に用意した．

一方，2015 年頃の SSL Humanoid の目指す最終形態を Figure 2 に示す．フィールドを斜め上から複数のカメラで撮影して，それら画像からフィールド内の各オブジェクトの位置や方向だけでなく，形状や姿勢を取得する．また，それらの視覚系はチーム間で共有する．3 次元の場合，視覚系のどのようなデータを共有するかによっていくつかの方法が考えられる．それらを，主に，参加チームの負担や自由度，競技運用側（共有視覚系を準備する側）の負担，配信するデータ量の観点から検討する．

2.1 カメラ画像の共有

フィールドの周りに設置されたカメラの画像を各チームへ配信を行う．

- 利点
  - チームは，独自の方法を試すことができ，様々な研究が展開できる．
  - 運用側の負担が小さい．
- 欠点
  - チームの負担が大きく画像処理が得意でないチームは参加しにくい．
  - 配信するデータ量が膨大になる．

2.2 ロボットの各部位の位置の共有

カメラ画像に基づき何らかの方法でロボットの各部位を識別し，それらの 3 次元的な位置や位置関係を各チームへ配信する．RoboCupSoccer シミュレーション 3D リーグではこの考えに似た方法が使われている．

- 利点
  - チームは，画像処理を一切しなくても構わないので，新規に参加しやすくなる．
  - 抽象化されたデータが手に入ることでロボットの運動制御や行動決定に専念できる．
  - 配信するデータ量が小さい．
- 欠点
  - 運用側の負担が大きい．
  - 安定した結果を得るには技術的な困難が大きい．
  - チームが 3 次元のデータ処理に独自の工夫をする余地がない．

2.3 ボクセルデータの共有

上述のカメラ画像を共有する方法では，チームの負担が大きすぎる．一方，ロボットの各部位の位置を共有する方法では，運用側の負担が大きすぎ，技術的な困難も予想される．そこで，二つの方法の中間的な位置付けとして，本稿では，フィールドの 3 次元空間のボクセルデータを共有することを提案する．

ここでボクセルデータとは，フィールド上の 3 次元空間を格子状に分割し，各要素ごとに物体があるかないかの 2 値を取るものとする．このようなデータは，各カメラ画像のシルエット画像に基づき，視体積交差法などの比較的単純な処理で得ることができる．また，同じ値が連続するデータであるので，圧縮により配信するデータ量を小さくすることができる．

ボクセルデータをどのように利用するかは，チームの自由であり，工夫のしどころである．これによって，認識の技術についても競うことができる．

この提案を実現するには，以下のようなことが求められる．

- 競技領域（約 3[m] × 4[m] × 0.5[m]）全てのデータが得られること．
- ロボットの形状を表現するために十分な解像度があること．身長 400[mm] 程度のロボットを表現するには，ボクセルの 1 辺は 10[mm] 以下であることが望まれる．
- 実時間で処理できること．目安として，1 秒間に 30 回以上の処理．
- 配信するデータ量が大きすぎないこと．

本稿の以下では，提案の方法を実現するために作成したプログラムについて説明し，上述の条件が満足できるか検証を行った結果について述べる．

3　3 次元形状の取得とデータ転送の手法

3.1　視体積交差法

本稿では 3 次元形状を得る方法として視体積交差法を用いる[保呂, 2006][ウ, 2001]．視体積交差法には大きく分けて Volume Intersection Method(VIM) と Space Carving Method(SCM) の 2 種類の方法がある．SCM はシルエット制約に基づいて，物体の内点を残して，それ以外の点を削り落とす (ボクセルカットと呼ぶ) 手法である．この手法では，ボクセルとピクセルの対応関係をあらかじめ計算しておくため，ボクセルカット時の計算量が少ないが，メモリを多く消費する．しかし，近年メモリが低価格になっており資源の制約は小さいので，本稿では SCM を利用する．

3.2　カメラキャリブレーション

複数のカメラの情報を統合するには，各カメラのパラメータを正確に得る必要がある．そこで，本稿では Z.Zhang のカメラキャリブレーションの方法を用いる[Zhang, 2000]．平面に貼り付けられたチェッカパターンを複数枚撮影し，そのパターンの交点の座標位置から，内部パラメータであるカメラの光軸中心 (c_x, c_y)[pixel]，焦点距離 (f_x, f_y)[pixel]，カメラのレンズ半径方向の歪み (k_1, k_2), 円周方向の歪み (p_1, p_2) を求める．そして，フィールド上の実座標と画像上の座標の組から外部パラメータである同次変換行列 $[R|t]$ を得る．

3.3　参照テーブルの作成

3.2 節より求められた内部パラメータおよび外部パラメータから，空間内のすべてのボクセルと各カメラの画像の画素との対応付けを行う．カメラの番号を c, ボクセルの中心座標を (x, y, z)[mm]，ボクセル番号を (i, j, k)，ボクセルのサイズを s[mm] とした場合，式 (1) の対応が得られる．

$$x = is, \quad y = js, \quad z = ks \tag{1}$$

フィールド座標系の (x, y, z) をカメラ座標系 (X, Y, Z) に変換する．式 (2)∼(9) ではカメラの番号 c を省く．

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = R \begin{pmatrix} x \\ y \\ z \end{pmatrix} + t \tag{2}$$

対応する撮像面上のピクセル番号 (u, v) を式 (3)∼(9) の手順で得る．

$$X' = X/Z \tag{3}$$

$$Y' = Y/Z \tag{4}$$

$$R = \sqrt{X'^2 + Y'^2} \tag{5}$$

$$X'' = X'(1 + k_1R^2 + k_2R^4) + 2p_1X'Y' + p_2(R^2 + 2X'^2) \tag{6}$$

$$Y'' = Y'(1 + k_1R^2 + k_2R^4) + p_1(R^2 + 2y'^2) + 2p_2X'Y' \tag{7}$$

$$u = f_xX'' + c_x \tag{8}$$

$$v = f_yY'' + c_y \tag{9}$$

以上の対応をすべてのカメラに対して，全てのボクセルについて求め，あらかじめメモリ上に数表として保持する．

$$\mathrm{table}_c(i, j, k) = (u, v) \tag{10}$$

3.4　背景差分法によるシルエット画像の生成

視体積交差法で用いるために，カメラ画像から対象のオブジェクトだけを抽出し 2 値化した画像をシルエット画像を生成する必要がある．カメラ画像と背景画像の画素値をそれぞれ $(R_{cuv}, G_{cuv}, B_{cuv})$, $(R'_{cuv}, G'_{cuv}, B'_{cuv})$，閾値を D_{th} とすると，シルエット画像の画素値 S_{cuv} を式 (11),(12) で得る．

$$D_{cuv} = |R_{cuv} - R'_{cuv}| + |G_{cuv} - G'_{cuv}| + |B_{cuv} - B'_{cuv}| \tag{11}$$

$$S_{cuv} = \begin{cases} 0 & (D_{cuv} > D_{th}) \\ 1 & (D_{cuv} \leq D_{th}) \end{cases} \tag{12}$$

3.5　ボクセルカット

ボクセルの値を $V_{ijk} \in \{0, 1\}$ とする．最初に全ボクセルに 1 をセットし，カメラごとにボクセルを走査し，表を引いて対応するシルエット画像の画素値が 1 であれば（シルエットでなければ），ボクセルをカット（値を 0）にする．一連の処理を C 言語風に表現すると以下のようになる．

```
for(i, j, k){
  V_ijk = 1;
}
for(c){
  for(i, j, k){
    if(V_ijk == 1){
      (u, v) = table_c(i, j, k);
      if(S_cuv == 1){
        V_ijk = 0;
      }
    }
  }
}
```

3.6　ランレングス圧縮

視体積交差法で得られた結果をクライアントへ送出するが，そのデータ量を減らすために，ランレングス圧縮を用

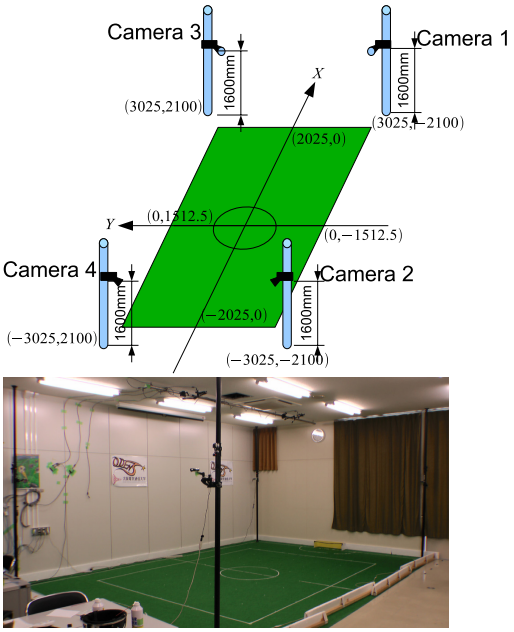

Figure 3: Experimental enviroment

Table 1: Specifications of experimental cameras

Company	Creative
Product name	Live!Cam NoteBook Ultra(VF0490)
Interface	USB2.0/1.1
Resolution	640×480

いる．1 個のランを 1[byte] で表し，上位 1[bit] でランの値，下位 7[bit] でランの長さとする．同一の値が 128 個以上の場合は 127 個ごとに分割する．

3.7 データ転送

クライアントへのデータ転送は UDP のマルチキャストを想定してる．毎回送信する内容はボクセルの 1 辺の長さ s[mm]，長方体領域の 3 辺の長さ [mm], ランレングス圧縮後のデータ量 [byte]，ランレングス圧縮後のデータである．

4 SSL Humanoid を想定した評価実験

4.1 実験の環境と装置

SSL Humanoid を想定して，対象とする空間を競技フィールド上の 3025[mm]×4050[mm]×500[mm] とする．Figure 3 に示すように，空間の周囲に斜め上から撮影する 4 台のカメラを設置する．用いたカメラの仕様を Table 1 に示す．

実験に用いたサーバ PC とクライアント PC の仕様を Table 2 に示す．2 台の PC はイーサネットで接続している．サーバ PC では，DirectShow を使って USB 接続されたカメラの画像を取り込む．現段階では，カメラ間で撮影の同期は取っていない．取り込まれた 4 台のカメラのシルエット画像に基づき，視体積交差法でボクセルデータを

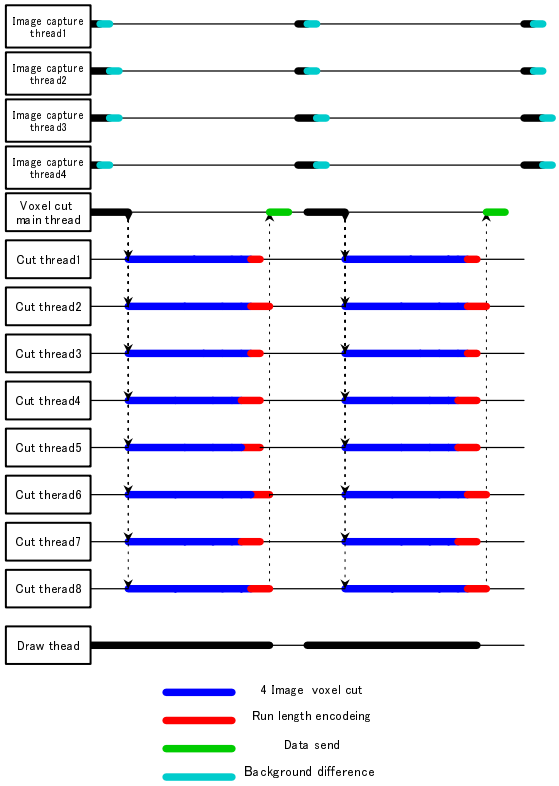

Figure 4: Timing chart

算出し，それをランレングス圧縮して送信する．クライアント PC は，受信したデータを復号し，3 次元表示する．

4.2 並列化

視体積交差法に用いたアルゴリズムは，ボクセルごとに独立しているので，並列化に向いている．そこで，空間を分割し，複数のスレッド処理することを試みる．全ての処理を 1 スレッド行う場合，1 回の処理に 55 ∼ 60[ms] かかっていたが，これが短縮されることが期待できる．サーバ PC の CPU は 4 コアで HTT により 8 スレッド同時実行が可能であるので，領域を 8 つに分割した．Figure 4 に各スレッドの処理のタイミングを示す．

4.3 評価方法

カメラの解像度を 640 × 480[pixel]，ボクセルの 1 辺の長さを 10[mm] に設定し，空間内に配置するオブジェクトを以下の 3 条件で変更し，1 回の処理に要する時間と，圧縮後のデータを計測する．

ex1 オブジェクトが 0 個

ex2 オブジェクトが 6 個（ヒト型ロボット 6 台）

ex3 ブジェクトが 10 個（ヒト型ロボット 6 台と紙筒 4 個）

ここで，ヒト型ロボットとして，身長 378[mm] の MANOI AT01 と身長 360mm の KTR-1HV を用いる．

Table 2: Specifications of experimental PCs

	Server PC	Client PC
Operating System	Windows 7 Professional 64bit	Windows Vista Home Premium 32bit SP2
Main Memory	3GByte	3GByte
Graphic Board	NVIDIA GeForce GTX260	Mobile Intel 4 Series Express Chipset
DirectX	DirectX 11	DirectX 11
CPU	Intel Core i7 950(3.19GHz)	Intel Core Duo U9400(1.4GHz)

Figure 5: Image of camera 1 (ex3)

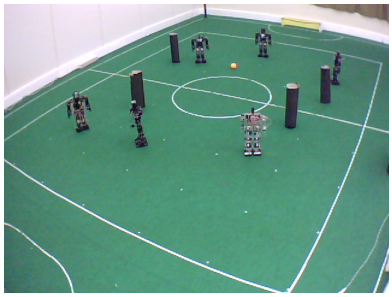

Figure 6: Image of camera 2 (ex3)

Figure 7: Image of camera 3 (ex3)

Figure 8: Image of camera 4 (ex3)

4.4 結果

ex3 の場合のカメラ画像の一例を Figure 5~8 に示す．また，これを処理して得られたボクセルデータの一例を Figure 9 と Figure 10 に示す．また，各条件において，1 周期あたりの処理時間を Figure 11 に示す．また，ex1, ex2, ex3 の各条件においてランレングス圧縮したデータ量の平均は，それぞれ 48526[byte], 57929[byte], 62374[byte] であった．

取得されたオブジェクトの一例として，ヒト型ロボットをフィールド中央に置いた場合の結果を Figure 12 に，コーナに置いた場合の結果を Figure 13 に示す．

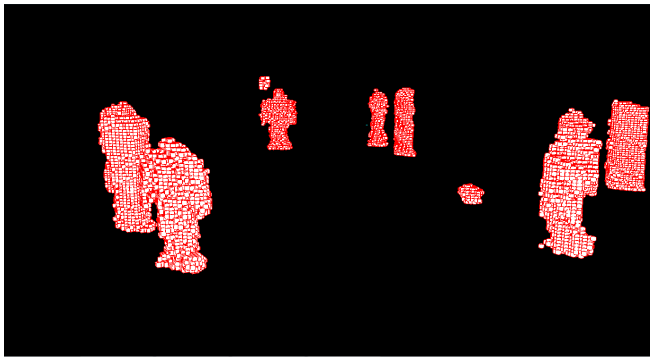

Figure 9: Result of voxel cut from lower view point (ex3)

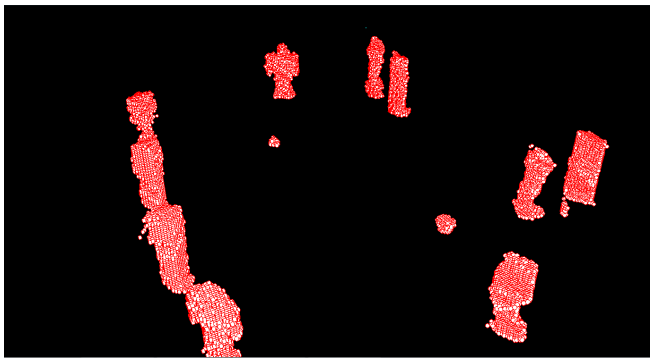

Figure 10: Result of voxel cut from higher view point (ex3)

4.5 考察

Figure 11 から，実装したプログラムでは，オブジェクトの数が増えるほど処理時間が増えることがわかる．SSL Humanoid の 2010 年のルールではフィールド上のロボット数は 6 台であるが，これが 10 台に増えても，1 秒間に 30 回の処理が可能である．これは，領域を分割し複数スレッドによる実行ですることによって，目標を達成できた．得られるボクセルのデータは，約 730[Kbyte] であるが，値が連続する部分が多いので，単純なランレングス

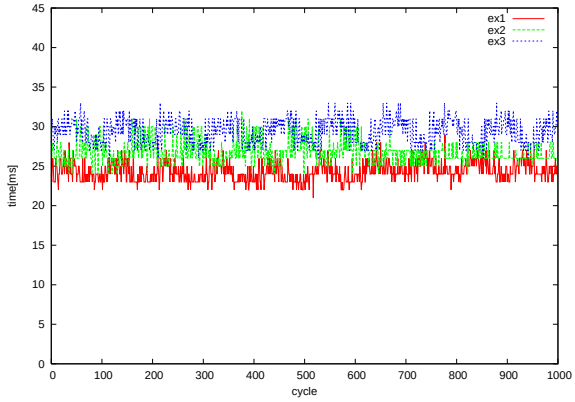

Figure 11: Processing time

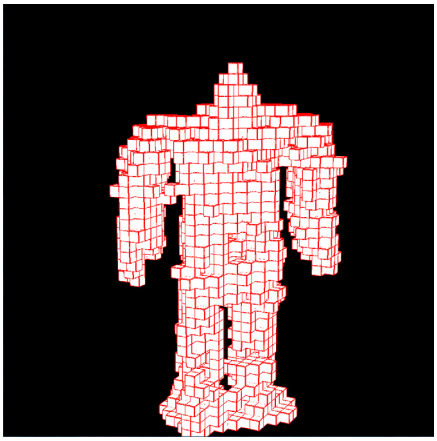
Figure 12: Voxels of humanoid robot at the center

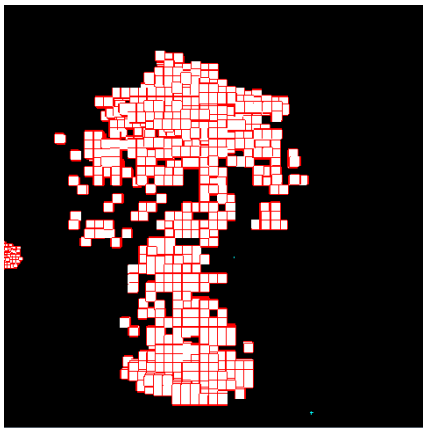
Figure 13: Voxels of humanoid robot at the corner

圧縮でも，1/10 以下の量にすることができた．これを 1 秒間に 30 回送るとすると，約 14[Mbps] であり，通常のネットワークで十分に可能である．

Figure 12 を見ると，1 辺 10[mm] のボクセルの解像度によって，身長 400[mm] 程度のヒト型ロボットの大まかな形状を表現できることがわかる．ただし，カメラの角度が悪いと，胴体と腕や左右の脚が分離できない場合がある．また，Figure 13 のような場合は，オブジェクトから最も離れたカメラの量子化誤差の影響で，必要以上にボクセルがカットされている．以上のような問題は，カメラの数を増やしたり，シルエット画像生成の処理を工夫することで，ある程度解決することが予想できるが，処理時間の増加が懸念される．

さらに，視体積交差法で複数のオブジェクトを対象とする場合，原理的に偽のオブジェクトが得られることがある．そこで，このような問題は，サーバの仕様と見なし，それを踏まえてチーム側が処理を工夫してもらうという考え方を提案したい．仕様の策定は，今後の課題である．

5 おわりに

RoboCup SSL Humanoid の共有視覚サーバを 3 次元化する方法として，競技フィールドの 3 次元空間のボクセルデータを各チームへ配信する方法を提案した．提案した方法を Intel Core i7 950 を搭載した PC に実装したところ，SSL Humanoid の競技フィールドを想定した空間を 1 辺 10[mm] のボクセルで切り出し，ランレングス圧縮してネットワークへ送り出す処理を 1 秒間に 30 回繰り返すことができた．また，圧縮したデータ量は，60[Kbyte] 程度であった．このことから，提案する方法が，現在の PC の性能で実現可能であることが明らかになった．

今後は，実時間で得られるボクセルデータに基づいて各ロボットの腕や脚の姿勢を認識し，それを利用してロボットの行動決定を行う研究やオブジェクトの抽出手法の改善を行う．

参考文献

[升谷, 2010] 升谷保博，成瀬正: SSLH: 外部カメラを用いたヒト型ロボットによるサッカー競技 RoboCup Soccer SSL Humanoid，人工知能学会誌, Vol.25, No.2, pp.213–219, 2010.

[Zickler, 2010] S.Zickler, T. Laue, O. Birbach, M. Wongphati, and M. Velosso: SSL-Vision: he Shared Vision System for the RoboCup Small Size League, RoboCup 2009, Robot Soccer World Cup XIII, Springer-Verlag,2010.

[保呂, 2006] 保呂毅: 視体積交差法: 視体積交差法を用いた実時間ポインティングジェスチャ認識, 日本機械学会ロボティクス・メカトロニクス講演会, 2006.

[ウ, 2001] ウ小軍，和田俊和, 東海彰吾, 松山隆司: 視体積交差法: 平面間透視投影を用いた並列視体積交差法, 情報処理学会論文誌, Vol.42, No.SIG6(CVIM2), 2001.

[Zhang, 2000] Z.Zhang: Calibration: A flexible new technique for camera calibration, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.22, No.11, pp.1330-1334, 2000.