データベースシステム 127－4
情 報 学 基 礎 67－4
（２００２．５．２１）

４ＤＶＭ：仮想環境への点検情報蓄積

喜多伸之、喜多泰代、楊海圏
産業技術総合研究所　知能システム研究部門

あらまし：観測データを長期に渡り保存し、後に検索できることは、原子力プラントの運転・保守の安全性向上に役立つ。しかしながら、そのための観測データは空間的・時間的に離散した膨大なデータであり、その蓄積・検索には、データ量の圧縮、データ間の関係の保存と利用、データの正規化など、解決すべき問題がある。我々はこれらの問題に対処するために、時間軸方向に拡張した仮想空間にデータを蓄積することを基本アイデア（**４ＤＶＭ（4 Dimensional Visible Memory）**）として提案している。本論では、４ＤＶＭの基本概念の説明と実現方法について述べたあと、移動ロボットにより収集した画像情報などを対象とした本手法の実装を紹介する。さらに、いくつかの実験例により本提案手法の有用性を示す。

4DVM：**Archiving Plant Inspection** Data **in a Virtual Environment**

Nobuyuki KITA, Yasuyo KITA, Hai-quan YANG
Intelligent Systems Research Institute
National Institute of Advanced Industrial Science and Technology

Abstract：The storage and later retrieval of observed data is an essential part of any support system aiding the safe operation and maintenance of a nuclear power plant. Archiving such data, however, which is broadly spread over space and time, is challenging: problems include how to reduce the amount of data, how to maintain relationships between data, how to normalize the data and so on. We propose a new concept, **4D Visible Memory**, whose basic idea is to store the data in a virtual space extended in the time dimension, as a solution. In this paper, a specific system which embodies the concept in the special case of archiving the image data gathered by mobile robots is introduced. Some experimental results show the actual usefulness of the concept.

1　はじめに

原子力プラントでは安全な運転・保守のために、多くのセンサーが常にプラントの状態を観測し様々なデータを収集している。そのようなデータを将来の閲覧のために蓄積することが商用プラントでも始められている。ただし、異なるセンサーで取得された情報は、個別に蓄積されるのが現状であり、後に各種データを関連付けて閲覧することは難しい。もし、これらのデータを関連付けて統合的に蓄積できれば非常に有用であるが、対象とするデータが空間的・時間的に膨大な範囲に広がっているためいくつかの解決すべき問題がある。具体的には、データ量（特に画像データの場合）がすぐに溢れる、種々のセンサーで収集した種類の異なるデータ間の関係を保存することが難しい、データの公正な比較のためには収集したデータを環境条件に関して正規化しておく必要がある、などである。

我々はこれらの問題に対処するために、時間軸方向に拡張した仮想空間にデータを蓄積することを基本アイデア（**４ＤＶＭ（4 Dimensional Visible Memory）**）として提案している。画像データを仮想物体表面に張りつける技術はテクスチャマッピングとして知られているが、それは仮想世界の画像を現実味高く合成するためのテクニックであり、データを蓄積するためのものではなかった。池内らは文化財を電子的に保存するために、仮想空間への実際の画像データのテクスチャマッピングを適用した[1]が、それはその瞬間の文化財の保存が目的であった。

これに対して、仮想空間に時間的な厚みを持たせ、時間的に変化するデータまでをも、テクスチャマッピングの要領で仮想空間に蓄積することにより、データ量や関係保存などの問題を解決しようというものである。

本論では、４ＤＶＭの基本概念の説明と核となる部分の実現方法について述べたあと、移動ロボットに搭載したセンサーにより収集した画像データを対象として本手法を実装したものを紹介し、さらに、いくつかの実験例により本提案手法の有用性を示す。

2　４ＤＶＭ：仮想環境への情報蓄積

時間軸方向に拡張した仮想空間にデータを蓄積することにより、時空間に離散した各種多様な情報を蓄積し、見たいときに見たいように可視化するアイデアを**４ＤＶＭ**（4 Dimensional Visible Memory）と名付けた。本章では４ＤＶＭの基本的な考え方である“仮想空間をデータの蓄積場とすること”の長所と、その実現手法について述べる。

2-1　データ蓄積場としての仮想空間

画像情報を始めとする種々のデータを蓄積する場とするアプローチには次のようなメリットがある。

・各種センサーにより１次元、あるいは、2 次元の形態で観測されたデータも、仮想空間に蓄積する際に、実世界の本来の 3 次元位置に還元して蓄積される。このことは、空間的・時間的に大きな圧縮効果をもたらし、さらに、自在な可視化を実現する。
・仮想環境技術として養われてきた多様なビジュアリゼーション技術が利用でき情報提示の自由度が拡大する。
・仮想環境技術として発展してきたハードウェア技術を用いて処理の高速化が計れる。

2-2　環境サーバー

４ＤＶＭを実現するには、

(1) データ蓄積の対象となる実際の環境とそこに存在するセンサーを計算機内において模擬し維持する機能。
(2) 様々なセンサーからデータを受け取ったり、可視化したデータを送出するための通信機能。
(3) 時空間的に離散したデータを仮想環境に蓄積する機能。

が必要である。上記(1)(2)に相当する機能は、実空間を模擬した仮想空間のみに存在する仮想物体の可視化を通して複数エージェント間での注意共有を実現するために開発した**環境サーバー**の基本機能としてすでに実現している[2]。したがって、環境サーバーに上記(3)に相当する次のような機能を付加することにより、４ＤＶＭ実現の核とする。

・入力データと仮想環境の位置合わせ機能
・データを収集した際の環境条件に関して正規化する機能
・データの仮想環境へのマッピング機能
・マッピングされた多様な情報の蓄積管理と検索機能
・蓄積されたデータの再マッピングによる可視化機能

図１は原子力プラント内の点検データの蓄積を対象とした場合の、環境サーバーとそれにアクセスするクライアント群の全

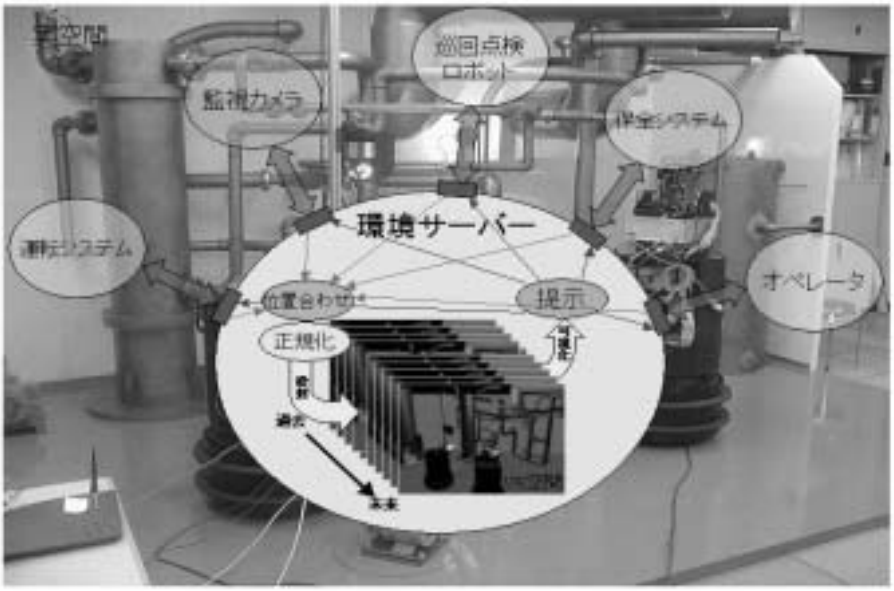

図 1　データ蓄積機能を付加した環境サーバーの概念図

容を模式的に示したものである。

3　画像データの蓄積と提示

3-1　環境サーバーの実装概要

移動ロボットに搭載したカメラで収集した画像データを仮想環境に蓄積するためには、まず、2 次元の入力画像と 3 次元の仮想環境を正確に位置あわせする必要がある。位置あわせができれば、画像から必要な部分を切り出して、対応する仮想物体表面に貼り付けることは容易である。仮想物体に貼り付けたデータは保存のために、関係データベースに貼り付けた場所などをインデックスとして蓄積する。蓄積したデータの可視化要求に対しては、必要なデータのみ関係データベースから検索し仮想環境にマッピングし、所望の視点・視線方向からレンダリングする。図 2 に現在の実装の概要を示す。クライアントからの蓄積・可視化要求は、いずれも環境サーバー本体が授受するため、クライアントからはシステム全体が一体として見えるが、システムの改変・拡張などを容易にするために、環境サーバー全体は実際には複数のモジュールにより構成している。具体的には、環境サーバー本体に対して、位置合わせ、マッピング、関係データベースの機能を、独立したバックエンドサーバーとして実現している。さらに、システム全体を汎用的に記述するために、個々のモジュール、および、モジュール間の通信の実装は汎用言語を用いて記述した。例えば、環境サーバー本体は、Java を基本として、仮想環境の記述であるシーングラフを維持したり画像を合成するためには Java3D、クライアントとのインタフェースには ORBacus（CORBA のフリー実装）を用いて開発している。

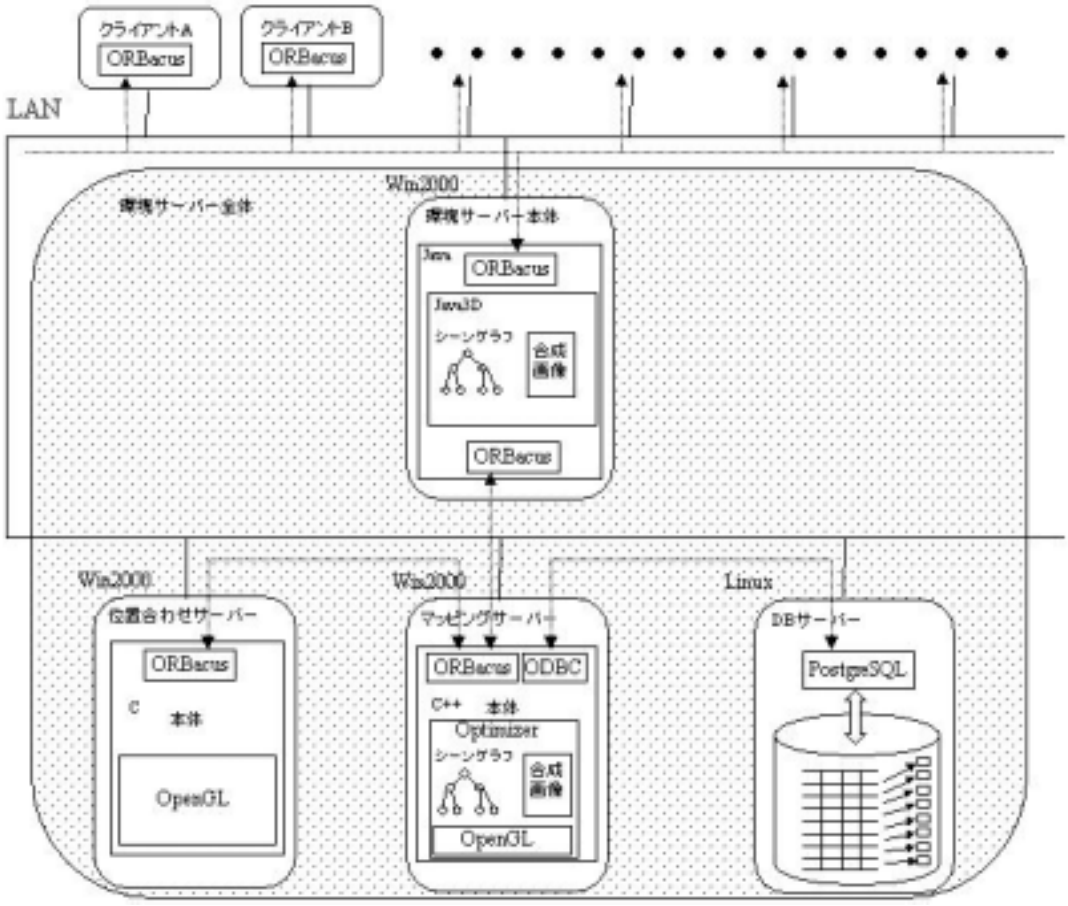

図 2　環境サーバーのシステム構成

3-2　位置合わせサーバーの実装

クライアントから環境サーバーに対して画像データの蓄積要求がくると、その画像を取得したカメラの初期パラメータ（視点位置や視線方向など）と画像データがマッピングサーバーを経由して位置合わせサーバーに送られる。このカメラの初期パラメータは、ロボットのオドメトリーや、3 次元位置姿勢センサーにより得られるが、通常少なからぬ誤差を含み、その結果、カメラの初期パラメータを用いて合成した仮想環境の画像と、入力画像との間にはずれを生じる。位置合わせサーバーのタスクは、このずれを解消するようカメラのパラメータを修正し、その修正したパラメータをマッピングサーバーに返すことである。

このような目的のために様々な手法が研究されてきたが、対象が曲面だけで構成されており、表面のテクスチャも使えず、さらに、それらの構造が複雑に込み入っている場合に適用できる手法はなかった。このような状況が原子力プラント内ではしばしば起こるので、我々は新たに遮蔽輪郭を用いた 3D-2D 位置合わせ手法を開発し用いている。詳しい手法の説明は参考文献[3]に譲るが、ここでは、処理結果の例のみ図 3 に示す。図の黒く塗りつぶした部分は仮想環境の投影であり、それを入力画像に重畳表示している。本例では、仮想環境の一部のみ用いている。図 3(a)でのずれが、位置あわせの結果図 3(b)のように解消されていることがわかる。

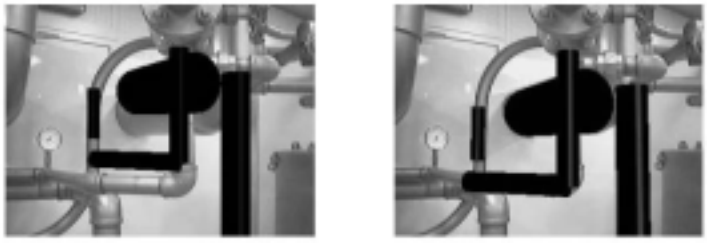

図 3　位置合わせの例

3-3　マッピングサーバーの実装

マッピングサーバーは、位置あわせのあと、入力画像から必要な部分を切り出し、仮想物体表面に貼りつける。

3.3.1　三角パッチへの分解

基本的には視野内に存在する仮想物体の輪郭に沿って画像を切り出し、物体表面に貼り付ければよい。ただし、表面の法線方向が観測方向と大きく異なる場合、貼り付けた情報の品質は悪くなる。したがって、同一対象にたいして多くの方向から観測した画像が入力されるような場合には、表面の法線方向により近い観測方向からの情報を採用してマッピングすることが重要である。

対象物体が平面で構成される場合は、平面ごとに採用すべき画像を決定すればよいが、曲面で構成される場合、同一曲面でも部位によって法線方向が大きく変わりうるので、それを単一表面として扱うことはできない。したがって、曲面は法線方向の変化が一定とみなせる程度におさまる小ささに分解する必要がある。ここでは、扱う曲面をパイプの円筒部分に限定し、一定の大きさの三角パッチに分解し、各三角パッチごとに画像情報をマッピングできるようにした。ただし、すべてのパイプを小さな三角パッチに分解するとシーングラフが巨大になり、記憶に必要なメモリ容量が急増するとともに、レンダリングの時間も極端に増加するので、情報の蓄積が必要となる可能性のある部分だけ三角パッチに分解している。

3.3.2　必要な情報の選択

入力画像から蓄積する必要のある部分のみ選択する。プラント点検の場合、巡回ロボットが自身で必要な情報を選択してそれを収集したり、オペレータが必要な情報を指示して提示を求める場合もあれば、他のクライアント、例えば劣化予測システムが必要な情報を指示し、それを巡回点検ロボットが収集したり、運転システムが必要な情報を指示して、それをオペレータに提示させる場合も想定される。このために、クライアントごとに異なる情報選択を可能にしたり、あるいは、その情報をクライアント間で共有できるメカニズムが必要である。注意制御のために環境サーバーに組み込んだ能動指標によりこれを実現する手法を検討しており、詳しくは文献[2]に述べている。

必要な情報の選択は、現在は、簡単のためにパイプを単位として行っている。例えば、パイプ５とパイプ６の画像情報が必要であると選択されているとすると、2 本のパイプを構成するすべての三角パッチがテクスチャマッピ

ングの対象となる。パイプの指定は、パイプ番号を陽に指定してもよいが、クライアントの視線をポインティングデバイスとして暗に指定することもできる。視野中心の感度が高いセンサー入力を蓄積したり、人間の目に提示する場合には便利な機能である。

3.3.3 テクスチャマッピング

指定されたパイプを構成するすべての三角パッチについて、入力画像において可視か否かを法線方向チェック、隠蔽チェックにより判定する。可視と判定された各三角パッチに対して、3 頂点の入力画像への投影位置を計算し、その 3 点を含む矩形領域を画像から切り出し、OpenGL の関数を用いて三角パッチにマッピングする。

図 4 はパイプ 13 について、1 枚の入力画像(a)から切り出した情報を(b)に貼り付けたところ(c)である。(d)はそれを違った角度から見たところ。

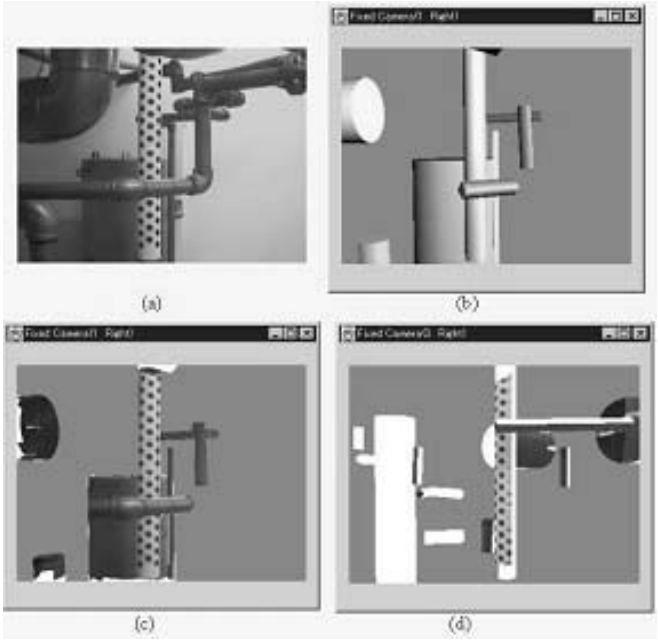

図 4　パイプへの画像データのマッピング

3.3.4 ワーキング DB への格納

三角パッチに対して切り取られた画像は、三角パッチ ID、入力日付、データの種類をインデックスとして、マッピングに必要な情報とともに一端、マッピングサーバー内のワーキング DB（WDB）に格納する。三角パッチ ID は OpenGL によりパイプに割り振られた番号と、分割プログラムが各三角パッチに割り振る番号から成る。データの種類はカラー画像、白黒画像、あるいは温度分布などである。

画像情報を WDB に格納する際、すでに同じ三角パッチ ID とデータ種類を持つレコードが存在した場合、それらは同一対象についての同種の情報であり、高い圧縮効果が見込める。特にプラントのような構造物の場合、表面の模様などが短時間に変化することはないので、1 回の巡回点検で得られた画像情報は本質的に同じとみなせる[1]。したがって、情報量が多い、つまり画像サイズが大きいほうのレコードを残し、小さいものは WDB から削除する

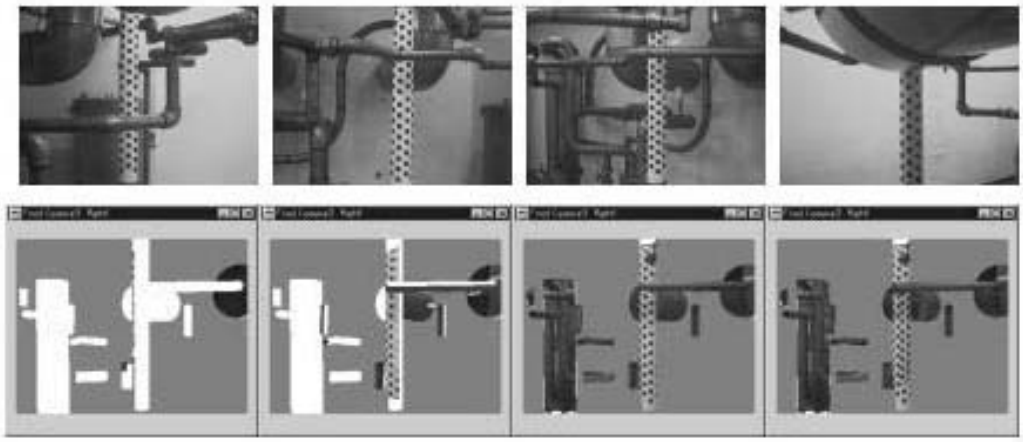

図 5　異なる観測方向の画像の圧縮

[1] ただし、このためには照明方向やカメラパラメータによる画像情報の変動を先に述べた正規化により完璧に取り除けている必要がある。

こととした。図 5 は、巡回点検時の入力画像を次々に WDB に蓄積したときの様子を示す。上段が入力画像であり、下段は WDB に格納されたテクスチャをマッピングして表示した結果である。異なる方向からの入力画像により、テクスチャを持つ三角パッチが増えていくことと、すでにマッピングしていた情報がより解像度の高い情報に置き換えられていくことがわかる。このようにして、観測方向が異なる入力画像についての情報圧縮を実現している。

3-4 データベースサーバーの実装

異なる時間の情報や、異なる種類の情報をテクスチャとして蓄積したり提示するためには、同じ場所に何枚ものテクスチャを貼り付けたり、それを切り替えてレンダリングできる必要がある。OpenGL や Java3D といった 3D グラフィックソフトウェアはテクスチャマッピング機能を持っているが、同じ場所に多重にテクスチャを貼ったり管理する機能はない。そこで、関係データベースサーバーをマッピングサーバーのバックエンドサーバーとし、WDB に格納した情報を情報入力の区切り（巡回点検の場合、一巡回の終了）において、マッピングサーバーに転送し永久保存する。提示の際には、マッピングサーバーからの要求により必要なテクスチャを検索し、マッピングサーバーに返す。マッピングサーバーはこれをシーングラフにマッピングしレンダリングする。

図 6 はマッピングサーバーが管理する関係 DB 内の保存レコード列の一部である。関係 DB の実装に用いた PostgreSQL では、画像のような大きなオブジェクトはポインタで管理し、実際の画像データはレコード列とは別の場所に格納されている。例の中に、同じ三角 ID とデータ種類を持つレコードが存在するが、それらは異なるラウンドの巡回点検中に得られた情報である。先にも述べたように、プラントの見えの変化は非常に緩慢であるので、これらの画像情報についても大きな圧縮効果が期待できる。ただし、現状では時間軸方向への画像の圧縮は行っていない。

Triangle ID	Input Date	Property	Mapped data
00100100	010701	color	* data
00100100	010701	intensity	* data
00100100	010701	friction	* data
00100100	010715	color	* data
00100100	010715	intensity	* data
00100100	010715	friction	* data
00100100	010801	color	* data
00100100	010801	intensity	* data
00100100	010801	friction	* data
00100101	010701	color	* data
00100101	010701	intensity	* data
00100101	010701	friction	* data

図 6　関係 DB 内の保存レコードの例

4　蓄積画像の提示実験

画像データの蓄積と提示機能を実装した環境サーバーについて実験を行なっている。画像データの蓄積については、位置合わせ手法の精度評価と改良を進めている段階であるので、ここでは、あらかじめ関係データベースに蓄積した画像データの提示実験を紹介する。

4.1 提示実験用蓄積画像

プラントモックアップ内の 2 箇所に固定したカメラにより図 7 に示す時系列画像を収集した。これら 2 台のカメラパラメータは既知である。画像は、パイプの表面に白色の木工用ボンドを塗布し、それが時間と共に乾いて透明に変化していく様子を 30 秒に 1 フレームの間隔で 85 枚撮影したものである。関係データベースに蓄積する際には、これをパイプ表面の亀裂状の模様が拡がって行く様子と見立てて、時間経過の方向を逆に、さらに、1 フレームの間隔を 1 日と見なして蓄積した。2 台のカメラは約 120 度異なる方向からパイプを観測したので、パイプ全周の 5/6 程度に画像データがマッピングされている。

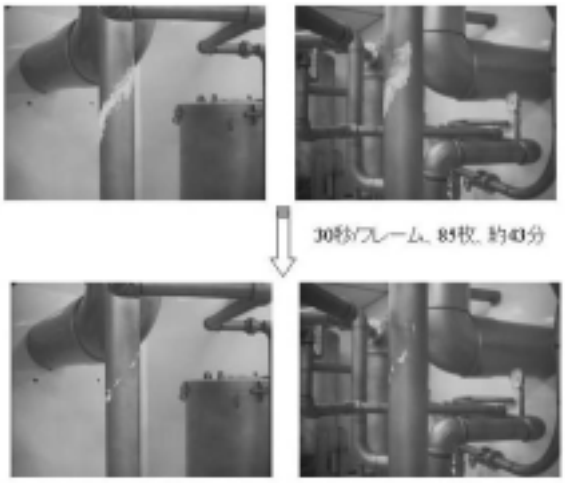

図 7　提示実験用入力画像列

4-2　提示実験

図 8 の右のウィンドウは提示実験用クライアントの GUI であり、パイプ 13 の 2000 年 3 月 15 日から 3 月 29 日の間に取得したカラーデータの中から適当なものを提示するよう要求しているところである。図 9 の左側の 2 つのウィンドウは、上が移動ロボットに搭載したステレオ能動視覚の左カメラに相当する仮想カメラ、下がオペレータが GUI を通じてコントロールしている仮想カメラ（GUI 上の鳥瞰図中の右中間あたりに浮いた状態で表示されている）のパラメータにしたがって環境サーバーがレンダリングした画像である。GUI ウィンドウの Continuous ボタンを ON にすると、関係 DB に蓄積している画像データを連続的に読み出しつつレンダリングできる。これと同時に仮想カメラの位置姿勢をコントロールすることにより、3 ヶ月弱の間のパイプ表面上の模様の変化を 30 秒程度に圧縮して、見たい方向から再現して見ることができた。

図 8　提示実験の画面例

4-3　情報提示処理の実装と実行時間

クライアントからの情報提示要求に対して環境サーバーは、必要なテクスチャ情報を DB から検索し、シーングラフにマッピングし、レンダリングするという処理を行う。時間軸方向に連続的な提示は、検索する情報の入力日付だけを変えながら、これら一連の処理を繰り返すことにより実現する。したがって、提示処理に要する時間が短ければ短いほど滑らかな情報提示が行える。

マッピングサーバーの初期の実装では、テクスチャのマッピングとレンダリングを Java3D により行なっていたため、PentiumIII デュアル PC でパイプ 1 本（三角パッチ数 459）だけを提示対象としたとき 1 フレームの提示に約 5 秒かかっていた。実装の様々な改良を施した結果、特に、テクスチャのマッピングとレンダリングを OpenGL

により実装したことにより、3フレーム/秒程度の表示が可能となり、スムースな連続表示が実現できている。

5 まとめ

多種多様な点検データを仮想環境に蓄積する新しいアプローチを提案し、移動ロボットにより収集する画像データを具体例としてシステムの実装を紹介した。改良を重ねたシステムにより、蓄積した画像データの提示実験を行なった結果、スムースで効果的な提示が行なえ、プラントの安全運転・保守に資する見通しを得た。しかしながら、本アプローチに基づくデータ蓄積によりどの程度のデータ圧縮が可能かについては未検討であり、さらに、三角パッチのサイズの最適化や、時間軸方向へのデータ圧縮など、データ圧縮のための改良点についても未着手である。また、提案システムの実用化のためには、データの正規化や注意制御、効果的な提示手法など、いくつかのチャレンジングな問題を解決する必要がある。

本論では、対象とする環境の幾何構造は変化しない前提で実装を行なったが、実際の原子力プラントでは定期点検の前後において、構造も含めた変化が起こり得る。したがって、幾何構造の変化にも対応できるようアプローチを改良する必要もある。

最後になったが、環境サーバーの構築に関して尽力いただいているカーネル株式会社の河村さんに感謝する。本研究は文科省の委託による原子力基盤クロスオーバー研究のもとでおこなった。

参考文献

[1] D. Miyazaki, T. Ooishi, T. Nishikawa, R. Sagawa, K. Nishino, T. Tomomatsu, Y. Takase, K. Ikeuchi：The Great Buddha Project: Modelling Cultural Heritage through Observation, Proc. of 6th Int. Conf. on Virtual Systems and Multimedia, pp.138-145 (2000).

[2] 喜多伸之：反射的・意図的注視制御の統合的な実現、信学論D-II, Vol.J84-D-II, No.8, 1701-1709(2001).

[3] 喜多泰代、喜多伸之：複雑環境中の遮蔽輪郭を利用した位置・姿勢検出, MIRU2000, II-43-48(2000).