JAERI-Tech
97-028

実例によるＡＶＳ利用手引

1997年7月

増子献児*・加藤克海*・久米悦雄・藤井　実

日本原子力研究所
Japan Atomic Energy Research Institute

本レポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所研究情報部研究情報課（〒319-11　茨城県那珂郡東海村）あて，お申し越しください。なお，このほかに財団法人原子力公済会資料センター（〒319-11　茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

This report is issued irregularly.
Inquiries about availability of the reports should be addressed to Research Information Division, Department of Intellectual Resources, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　㈱高野高速印刷

JAERI-Tech 97-028

実例によるＡＶＳ利用手引

日本原子力研究所計算科学技術推進センター
増子　献児*・加藤　克海*・久米　悦雄・藤井　　実

（１９９７年６月２日受理）

日本原子力研究所では，可視化ソフトウェアＡＶＳについて原研の一般のユーザが簡単に理解，利用できるようにＡＶＳの原研版利用手引を作成している。本報告書では「避難シミュレーション」を題材として，シミュレーション結果のＡＶＳを用いた可視化及びその可視化された画像のＶＴＲへの録画作業について解説する。本報告書の特長は，読者が上記の題材を本報告書に従ってＡＶＳで処理することで，ＡＶＳ利用に必要な一連の作業手順が修得できるようになっていることである。

東海研究所：〒319-11　茨城県那珂郡東海村白方白根2-4
＊　高度情報科学技術機構

AVS User's Guide on the Basis of Practice

Kenji MASUKO*, Katsumi KATO*
Etsuo KUME and Minoru FUJII

Center for Promotion of Computational Science and Engineering
Japan Atomic Energy Research Institute
Tokai-mura, Naka-gun, Ibaraki-ken

(Received June 2, 1997)

The special guides for the use of visualization software AVS have been developed at Japan Atomic Energy Research Institute(JAERI). The purpose of these guides is to help the AVS users understand easily the use of the one, due to the fact that it is so difficult for beginners to understand the original manuals.

In this report, "Transportation Evacuation Simulation" is taken up as an object of visualization, and the procedure of visualization and images recording by using the AVS are described.

By using the AVS according to this report, a series of the procedure which are necessary for use of the AVS can be acquired.

Keywords : AVS (Application Visualization System), Visualization, Animation, Image, Transportation Evacuation Simulation.

* RIST (Research Organization for Information Science Technology)

目　　次

1．　はじめに …… 1
2．　作業概要 …… 2
3．　実例の題材について …… 3
4．　入力データの作成 …… 5
　4．1　ＡＶＳデータ …… 6
　4．2　変換プログラムによる入力データの作成 …… 9
5．　可視化プログラム（ＡＶＳネットワーク）の作成 …… 19
　5．1　ＡＶＳネットワーク作成の特徴 …… 19
　5．2　ＡＶＳネットワークの作成の準備 …… 22
　5．3　ＡＶＳネットワークの作成例 …… 24
　5．4　モジュール作成 …… 28
6．　画像の保存 …… 31
　6．1　アニメーション作成の概要 …… 32
　6．2　システム構成 …… 33
　6．3　アニメーション作成の作業 …… 35
7．　おわりに …… 37
謝　辞 …… 37
参考文献 …… 37

Contents

1. Introduction 1
2. Outline of Visualization Procedure 2
3. Object of Visualization 3
4. Input Data for AVS 5
4.1 AVS-data 6
4.2 Data Conversion to AVS-data 9
5. Visualization Program (AVS-network) 19
5.1 Features of AVS-network 19
5.2 Preprocess of AVS-network 22
5.3 Example of AVS-network 24
5.4 Generation of Module 28
6. Images Recording 31
6.1 Outline of Making Animation 32
6.2 Configuration of Visualization System 33
6.3 Making of Animation 35
7. Concluding Remarks 37
Acknowledgments 37
References 37

1　はじめに

計算科学において、シミュレーション計算で出力する数値データを可視化することは、計算結果を理解したり判断したりする手段として非常に重要なものである。近年、可視化は、ハードウェアの高速化及び低価格化、ネットワーク技術の進歩などに伴い、より身近かなものになってきた。そのため、新しい可視化ソフトウェアも次々と開発されるようになってきている。

ＡＶＳ社の開発した汎用可視化ツールＡＶＳは、現在の所、スーパーコンピュータからＷＳ、ＰＣに至る広い範囲で最も幅広く使用される可視化ソフトウェアである。このＡＶＳの特長は、マウス操作によって可視化処理の部品を組み合わせて簡単に可視化できることである。しかしながら、このＡＶＳのメーカマニュアルは一般ユーザにとっては非常に読みにくく、画像処理の初心者がこれを使いこなすためには、メーカマニュアルとかなり格闘せざるを得ない部分がある。

そこで、日本原子力研究所（以下、原研）では、原研の一般ユーザがこのＡＶＳを簡単に理解、利用できるようにするためにＡＶＳの原研版利用手引を作成している。平成７年度においては「ＡＶＳ利用手引書」[1] を作成し、主にＡＶＳを利用したことがないユーザを対象として、ＡＶＳの操作手順について解説した。しかし、この利用手引は、具体的な題材を使用していないので、ＡＶＳを実際に使っていく上で、まだとっつきにくい面があった。このため、平成８年度は、実際のシミュレーション結果を対象にＡＶＳによる可視化の実例をまとめた。本報告書では、題材として「避難シミュレーション」[4] を取り上げ、シミュレーションの結果をＡＶＳに読み込んで可視化し、その画像をＶＴＲに記録するまでの作業について解説する。

第２章で実際に行った作業の概要について述べ、第３章で実例の題材である避難シミュレーションの概要及びそのシミュレーション結果をどのように表現するのかについて述べる。第４章から第６章では、各作業（入力データの作成、可視化プログラムの作成、可視化結果の保存）の詳細について述べる。なお、本作業では東海研計算センターの画像処理サーバ及び動画像処理システムを利用している。

また、本報告書を読み進めていく上で「ＡＶＳ利用手引書」[1] を参考文献としてはぜひ手元に置いておくことを推奨する。さらに、「動画像システムの開発」[2] 及び「リアルタイムモニターシステムの開発」[3] も合わせて準備しておくことが望ましい。

2　作業概要

本報告書では題材として「避難シミュレーション結果の可視化」を取り上げ、シミュレーションの結果をＡＶＳに読み込んで可視化し、画像をＶＴＲに記録するまでの作業について解説します。（図 2-1を参照）

1. 入力データの作成
   可視化に必要な数値データをＡＶＳで処理できる形式に変換します。
   実例では道路の混雑を表すデータ（交通密度データ）と道路網の構造をいくつかのリンクで定義したデータ（リンクの位置情報、リンクの接続情報）を変換プログラムを用いてＡＶＳで処理できる形式に変換しています。

2. 可視化プログラム（ＡＶＳネットワーク）の作成
   ＡＶＳでは、モジュール（可視化の一部分の処理を行なうプログラム）を組み合わせて可視化プログラムを作成します。ＡＶＳで作成した可視化プログラムをＡＶＳネットワークといいます。
   実例で作成するＡＶＳネットワークは交通密度データ及び道路網の構造を定義したデータを読み込み、各リンクを走行している車両数を色の変化に置き換えた画像として表示します。

3. 画像の保存（アニメーションの作成）
   ここでは、可視化した結果を用いて、再検討やプレゼンテーションに利用するために、ディスプレイに表示した画像を保存する作業を行ないます。実例では、可視化した結果をアニメーションとしてビデオテープに保存します。

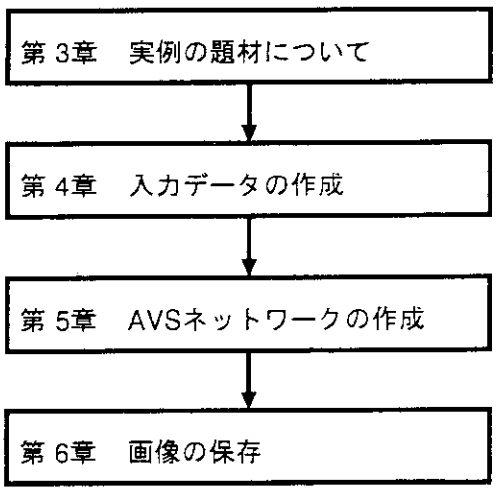

図 2-1　本手引書の構成

3　実例の題材について

本章では、実例の題材である避難シミュレーションについて説明します。 続いて避難シミュレーションの結果をどのように可視化するのかを説明します。

避難シミュレーションとは、ある地域に緊急事態が発生して避難の必要性が生じた場合の住民の避難状況を模擬するものです。住民は避難の手段として車両を利用するものとし、道路上を走行している車両数を以下に述べるような数値データ（交通密度データ）で出力します。

まず、対象となる地域の道路網の構造を適当な区間毎に分割したリンクで構成して、この情報を道路網の構造を定義したデータとして避難シミュレーションプログラムに読み込んで実行します。避難シミュレーションプログラムは一定の間隔（１０秒）毎に、各リンク上に存在する車両数を計算し、数値データ（交通密度データ）としてファイルに出力します。交通密度データとは、道路１ km 当たりに存在する車両台数を表しています。シミュレーション時間が経過するに伴い交通密度データが刻々と変化していきます。交通密度データを追っていくと、任意の時刻における道路の混雑状況を把握することができます。また、その変化に注目すれば、車両が避難していく様子を知ることができます。

避難シミュレーションの結果を可視化するためには以下のような作業を行います。

最初に避難シミュレーションがある時刻に出力する交通密度データをカラーデータに変換して各リンク毎に走行している車両数を色によって表します。このために以下の処理を行ないます。また、その処理過程を図 3-1に示します。

1. 道路網に関するデータと各リンクの交通密度データをＡＶＳに読み込みます。
   （第 4章 入力データの作成で解説します）

2. 交通密度データの大きさに対応する色を定め、それに従って交通密度データをカラーデータに変換します。道路網の構造を定義したデータからディスプレイに道路網を表示し、各リンクをカラーデータに従って色づけして表示します。
   （第 5章 可視化プログラムの作成で解説します）

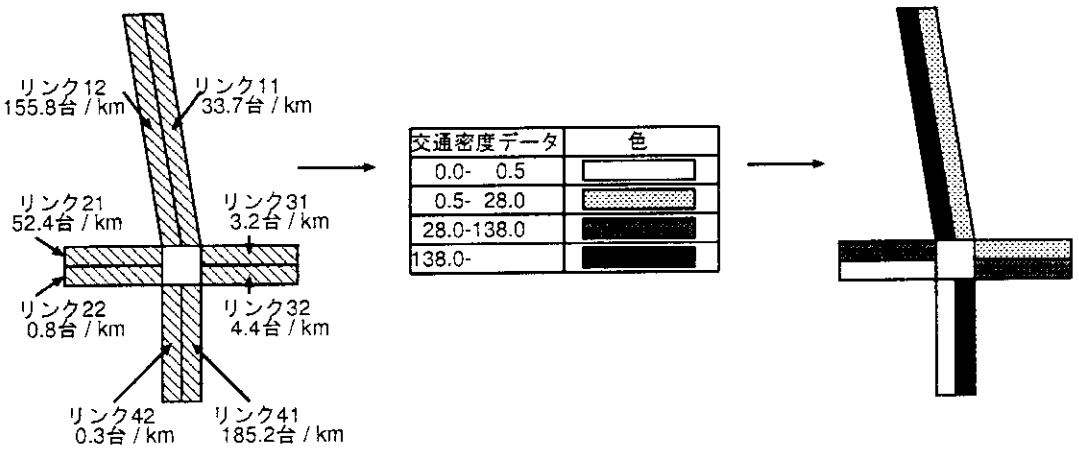

図 3-1　交通密度データの可視化

次に交通密度データの変化をアニメーションによって表します。アニメーションは、避難開始から一定時間毎に出力される交通密度データを連続的に可視化してビデオテープに画像をコマ撮りする方法によって作成します。アニメーションの作成の処理過程を図3-2に示します。
（第6章 画像の保存で解説します）

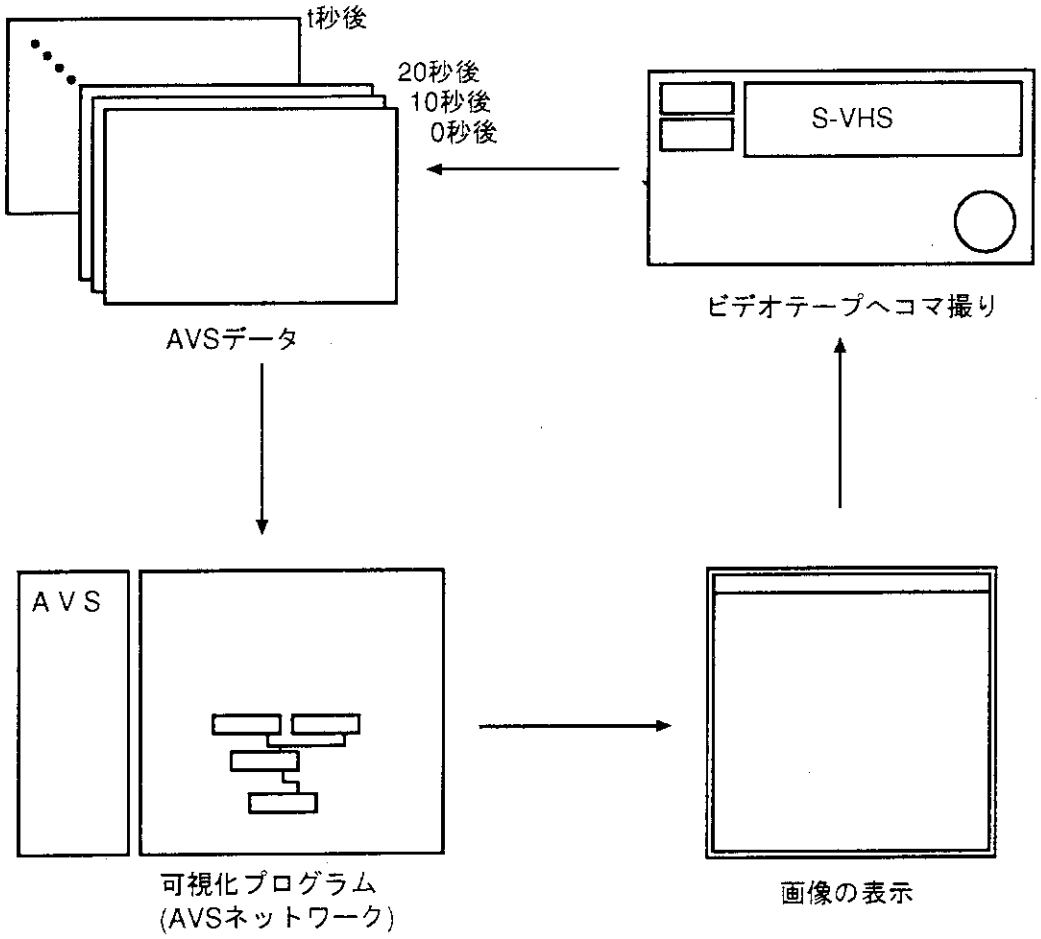

図 3-2　アニメーションの作成

4　入力データの作成

本章では、ＡＶＳに読む込む入力データを作成する作業について説明します。入力データの作成では、データをＡＶＳで処理できる形式（以下、ＡＶＳデータ）に変換します。

まず、本作業を行なうために必要な知識（ＡＶＳデータのタイプ、ＡＶＳデータへの変換方法）について説明します。次に、実例で行なった作業に沿って入力データの作成について説明します。図 4-1に第 4章の構成を示します。

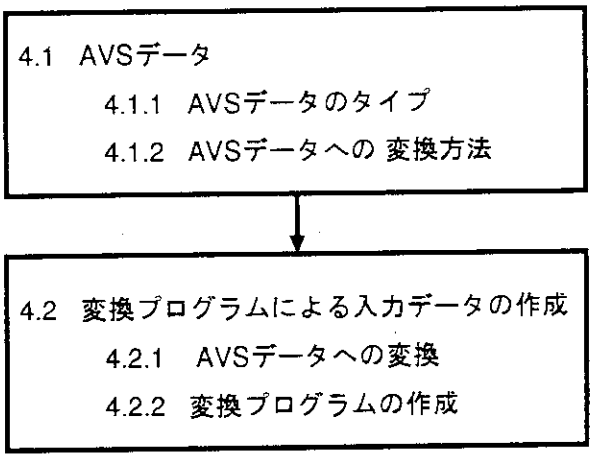

図 4-1　第 4 章の構成

4.1 ＡＶＳデータ

入力データの作成では、最初にどのタイプのデータに変換してＡＶＳに読み込むのか、次にどのような方法でＡＶＳデータに変換するのかを検討する必要があります。本節ではこれらを考えるのに必要な知識としてＡＶＳデータのタイプ、ＡＶＳデータへの変換方法及びＡＶＳへの読み込み方法について解説します。

4.1.1 ＡＶＳデータのタイプ

以下に主にＡＶＳで用いられるデータのタイプとその特徴について説明します。元となるデータの特徴や性質などを考慮してどのタイプに変換するのかを決定します。

- field データ（[1] p53 参照）

  配列構造（構造化データ）を一般化したデータタイプです。データを格納する格子点をノードといい、配列のインデックスがノードの並びを示します。ノードが格子状に並ぶのでデータのサンプリングに便利です。ＡＶＳの配列は単一の値（スカラーデータ）又は複数の値（ベクトルデータ）を持つことができます。さらに各ノードは座標データを持つことができます。座標データはフィールドの要素をマッピングする物理空間の座標を示します。field データは座標データの持ち方によって uniform,rectilinear,irregular の 3 つに分類されます。

- ucd データ

  図 4-2に示すような 8 種類の構造を持つセルを組み合わせた非構造格子を持つデータタイプです。 各セルはノードとそのセルを構成するノードの結合情報を持ち、各ノードの座標データ、ノード番号及びセル番号を属性データとして持ちます。 ノードは複数のセルで共有することができます。データは各ノード、各セル及びモデル（構造全体）に、それぞれ、ノードデータ、セルデータ及びモデルデータとして格納します。データはスカラーデータ及びベクトルデータを混在させることができます。ucd データによる可視化には、ノードデータをもとに処理する方法とセルデータをもとに処理する方法があります。また、モデルとはセルを組み合わせた構造全体をいいます。モデルについてはノードやセルと同様にデータを格納することはできますが、モデルデータを処理するモジュールがないために通常は使用しません。実例では、道路網の構造を line セルを組み合わせて定義します。またセルデータとして各リンクの交通密度データを格納します。

- geometry データ（[1] p67 参照）

  ＡＶＳで扱う幾何データです。幾何形状をプリミティブ（Polygon, Line, Mesh, Sphere, Polytriangle）の組合せで表します。geometry データは、形状を表すデータに加えて色、法線、表面特性などの属性データを持ちます。

  また、field データ又は ucd データとしてＡＶＳに取り込んだデータは最終的には geometry データに変換され、さらにレンダリング処理によってイメージデータに変換されて画像としてディスプレイに表示されます。

- primitive データ

  一般的に使用される整数、浮動小数点、バイト、文字列などのデータを指します。primitive データは主にモジュールのパラメータとして使用されます。

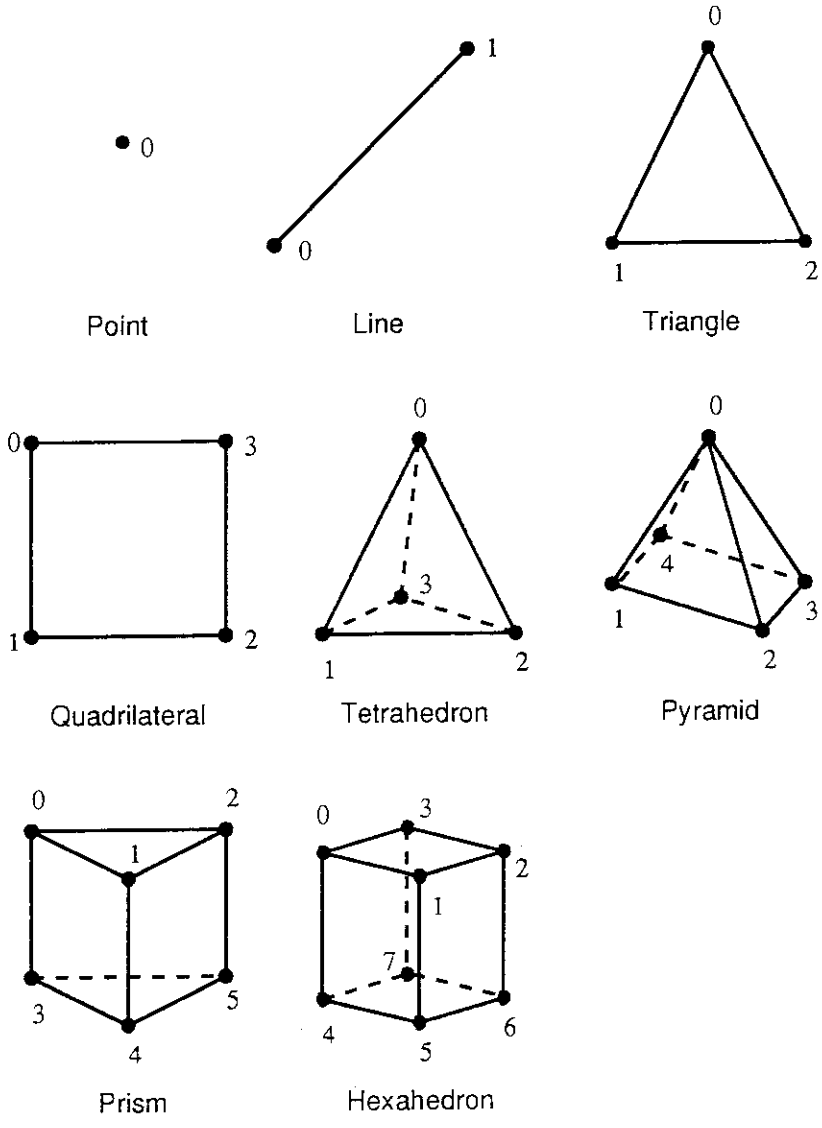
0
Point
1
0
Line
0
1
2
Triangle
0
3
1
2
Quadrilateral
0
3
1
2
Tetrahedron
0
4
1
3
2
Pyramid
0
2
1
3
5
4
Prism
0
3
2
1
7
4
5
6
Hexahedron

図 4-2　セル

4.1.2 ＡＶＳデータへの変換及びＡＶＳへの読み込み方法

以下に、ＡＶＳデータへの変換及びＡＶＳへの読み込み方法について説明します。

- 方法１ 変換プログラムによる変換
  変換プログラムを作成して、あらかじめ可視化に必要なデータをＡＶＳデータに変換する方法です。変換プログラムはＣやＦＯＲＴＲＡＮなどのプログラミング言語を使用して作成します。この方法は主に可視化に必要なデータを ucd データ又は field データとしてＡＶＳに読み込む場合に使用します。ＡＶＳのフォーマットで記述したファイルはＡＶＳ標準のモジュール (read field, read ucd) で読み込むことができます。ucd データのファイルフォーマットについては次節で説明します。また field データのファイルフォーマットについては [1] p57 を参照して下さい。

- 方法２ 読み込み用のモジュール作成による変換
  データをＡＶＳデータとして読み込むための新しいモジュールを作成する方法です。まず Module Generator モジュールによりモジュールの骨格部分のソースプログラムを作成します。次にソースプログラムにＣ, ＦＯＲＴＲＡＮなどのプログラミング言語を使用して処理ルーチンを加えます。処理ルーチンには生データをＡＶＳの構造体に読み込むルーチンを記述します。最後にコンパイルにより実行ファイルを生成してＡＶＳに取り込みます。モジュールの作成方法については「5.4 モジュール作成」及び [1] p92 を参照して下さい。

- 方法３ その他の変換方法
  上記の方法以外にＡＶＳにデータを読み込むには以下のような方法があります。

  - アスキー記述ファイルを利用する方法 ([1] p64 参照)
    field データとして読み込む場合に使用する方法です。テキスト形式、FORTRAN の書式なし、全てバイナリなどで記述された元となるデータの参照方法をアスキー記述ファイルとよばれるファイルに記述します。アスキー記述ファイルを read field モジュールで読み込むと記述内容に従ってデータをＡＶＳに読み込むことができます。

  - file descripter モジュールを利用する方法
    field データを読み込む場合に使用する方法です。この方法では、file descripter モジュールでデータの参照方法を指定します。データの参照方法はフォームファイルに保存します。フォームファイルは data descriptor モジュールにより読み込んでそれに従ってＡＶＳにデータを読み込みます。

  - フィルタープログラムを利用する方法 ([1] p67 参照)
    geometry データを読み込む場合に使用する方法です。指定のフォーマットで記述したデータをフィルターによって変換して geometry データを生成します。

4.2 変換プログラムによる入力データの作成

本節では、変換プログラムによる入力データの作成について解説します。この方法では変換プログラムをCやＦＯＲＴＲＡＮなどのプログラミング言語により作成し、このプログラムを用いて元のデータをＡＶＳデータに変換します。

実例では、変換プログラムを用いて避難シミュレーションのデータ（道路網の構造を定義したデータ、交通密度データ）を ucd データに変換しています。

まず、ucd データへの変換及び ucd データのファイルフォーマットについて説明します。次に、実例に沿って変換プログラムの作成について説明します。

4.2.1 ucd データへの変換

ここでは、簡単な例を挙げてどのようにして避難シミュレーションのデータを ucd データに変換するのかについて説明します。図 4-3に示す交差点において避難シミュレーションを実行した場合を考えます。道路網の構造を定義したデータを入力とし、避難シミュレーションを実行すると、各リンクの交通密度データが出力されます。

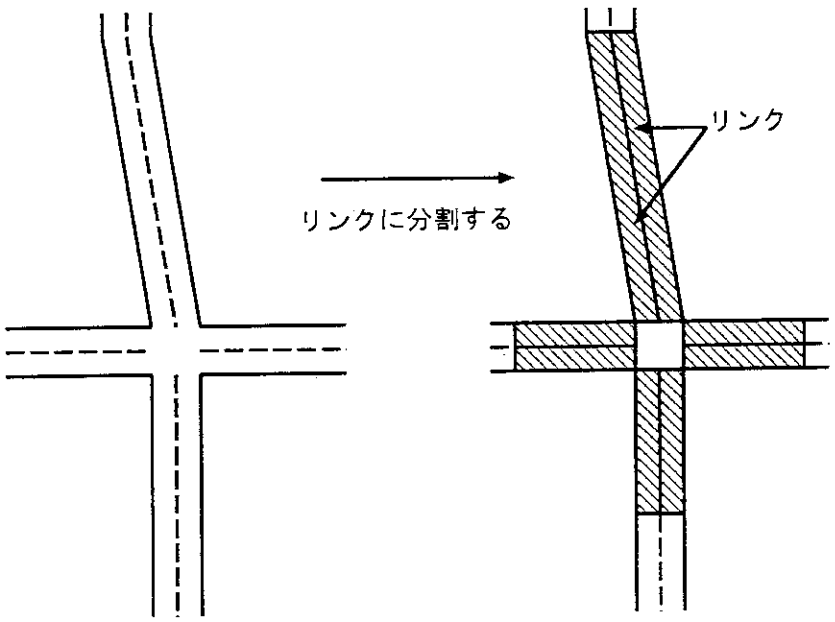

図 4-3　交差点

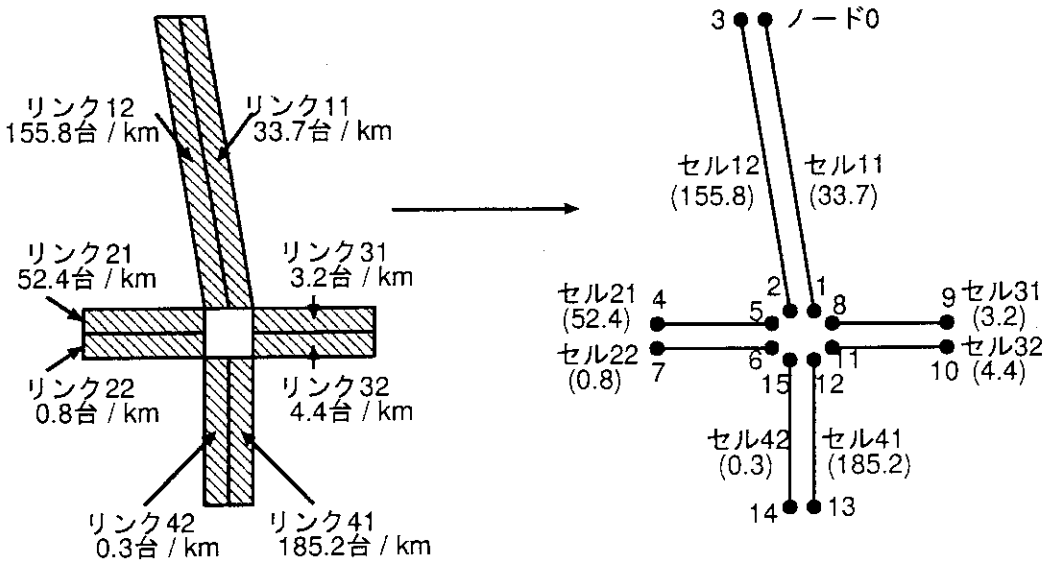

図 4-4　セルへの置き換え

これらの避難シミュレーションのデータを ucd データに変換する手順は以下のようになります。(図 4-4参照)

- 各リンクを line セルで置き換える。
- 各 line セルを構成するノードにノード番号を割り振る。
- 各 line セルにセル番号を割り振る。
- 各リンクの交通密度データをセルデータとして与える。

ucd データのファイルフォーマットは、アスキーフォーマットとバイナリフォーマットがあり、いずれの場合にもファイル名に拡張子.inp を付けます。実例ではアスキーフォーマットを使用しています。アスキーフォーマットはデータをファイルにテキスト形式で記述します。図 4-5に ucd データのアスキーフォーマットを示します。ただし、コロン (:) 及びその前の数字は、説明のためにつけた行番号であり、実際には記述しません。

また、図 4-3の交差点に関するデータをアスキーフォーマットに従って ucd データに変換した例を図 4-6に示します。図 4-3の交差点のモデルでは、セルデータ（交通密度データ）のみを持つためにノード及びモデルに関するデータの記述はしていません。

```
 1: ノード数 セル数 各ノードのデータ数 各セルのデータ数 モデルのデータ数
 2: ノード番号 X 座標 Y 座標 Z 座標
 3: ノード番号 X 座標 Y 座標 Z 座標
 4: ノード番号 X 座標 Y 座標 Z 座標
 5: ........ ... ... ...
 6: セル番号 材料番号 セルタイプ セルを構成するノードの並び
 7: セル番号 材料番号 セルタイプ セルを構成するノードの並び
 8: セル番号 材料番号 セルタイプ セルを構成するノードの並び
 9: ...... ...... ........ ........................
10: ノードのデータ成分の種類 各種類のベクトル長 各種類のベクトル長 ...
11: 各種類のラベル, 単位
12: 各種類のラベル, 単位
13: ................... ..
14: ノード番号 ノードデータ ノードデータ ...
15: ノード番号 ノードデータ ノードデータ ...
16: ノード番号 ノードデータ ノードデータ ...
17: ........ ........... ........... ...
18: セルのデータ成分の種類 各種類のベクトル長 各種類のベクトル長 ...
19: 各種類のラベル, 単位
20: 各種類のラベル, 単位
21: ............. ...
22: セル番号 セルデータ セルデータ セルデータ ...
23: セル番号 セルデータ セルデータ セルデータ ...
24: セル番号 セルデータ セルデータ セルデータ ...
25: ...... ........ ........ ........ ...
26: モデルのデータ成分の種類 各種類のベクトル長 各種類のベクトル長 ...
27: 各種類のラベル, 単位
28: 各種類のラベル, 単位
29: ............. ...
30: モデル番号 モデルデータ モデルデータ モデルデータ ...
31: ........ ........... ........... ........... ...
```

図 4-5　ucd データのアスキーフォーマット

- 1: （1行目）

モデルを構成するノードの総数、セルの総数、各ノードの持つデータ数、各セルの持つデータ数、モデルが持つデータ数を記述します。図 4-3の交差点ではモデルを１６個のノードと８つの line セルで構成して、交通密度データをセルデータとして格納しています。モデルを構成するノードの総数を１６、セルの総数を８、各ノードの持つデータ数を０、各セルの持つデータ数を１、モデルが持つデータ数を０とします。(図 4-6 1:(１行目))

- 2: （2行目）～5: （5行目）

  モデルを構成するノードのノード番号及びその座標（3次元上の座標）を記述します。ノード番号は各ノードを区別できるような整数値を記述します。ノードは3次元の座標を持ちます。避難シミュレーションの入力データからはx,y座標しか得られないのでz座標として適当な値（ここでは0.0）を与えます。図4-3の交差点では、図4-4で各ノードに割り振ったノード番号とその座標を記述します。(図4-6 2:(2行目)～17:(17行目))

- 6: （6行目）～9: （9行目）

  モデルを構成するセルのセル番号、材料番号、セルタイプ、セルを構成するノードの並びを記述します。セル番号は各セルを区別できるような整数値を記述します。 材料番号は、モデル全体を複数のモデルとして扱う場合に使用しますが、モデルデータを処理するモジュールがないために通常はすべて1とします。セルタイプは、その種類によって以下の表4-1に従って記述します。

表4-1　ucdデータのセルタイプ

セルのタイプ	記述
Point セル	pt
Line セル	line
Triangle セル	tri
Quadrilateral セル	quad
Tetrahedron セル	tet
Pyramid セル	pyr
Prism セル	prism
Hexahedron セル	hex

図4-3の交差点では、セル番号は図4-4で各セルに割り振ったセル番号を記述します。セルタイプはlineを記述して、セルを構成するノードの並びは図4-4に従いlineセルの両端のノード番号を記述します。(図4-6 18:(18行目)～25:(25行目))

- 10: （10行目）

  ノード（各セル, 各モデル）の持つデータの種類及び各データのベクトル長を記述します。ucdデータでは、fieldデータと同様に各ノード（ucdデータの場合は各セル, 各モデルも同様）に複数のデータ（スカラーデータ、ベクトルデータ）を混在して格納することができます。

- 11: （11行目）～13: （13行目）

  ノードに格納するデータのラベルと単位を記述します。交差点のモデルでは、ラベルに交通密度データ、単位に台／kmと記述します。

- 14: （14行目）～17: （17行目）

  ノード毎に、ノード番号とノードに格納するデータを記述します。
  ここでは記述しません。

- 18: （１８行目）～30: （３０行目）

ノードと同様にセル及びモデルに関する情報を記述します。図4-3の交差点では各セルに交通密度データ（スカラーデータ）を格納するので、セルデータの種類を１、セルデータのベクトル長を１とします。(図4-6 26:(２６行目)～27:(２７行目)) また、セル毎にセル番号及びセルデータ（交通密度データ）を記述します。(図4-6 28:(２８行目)～35:(３５行目))

```
 1: 16 8 0 1 0
 2: 0 -2.509 0.397 0.000
 3: 1 -2.623 0.217 0.000
 4: 2 -2.640 0.228 0.000
 5: 3 -2.526 0.408 0.000
 6: 4 -3.102 0.364 0.000
 7: 5 -2.629 0.232 0.000
 8: 6 -2.634 0.213 0.000
 9: 7 -3.107 0.345 0.000
10: 8 -2.627 0.231 0.000
11: 9 -2.285 0.059 0.000
12: 10 -2.294 0.042 0.000
13: 11 -2.636 0.214 0.000
14: 12 -2.624 0.216 0.000
15: 13 -2.709 0.111 0.000
16: 14 -2.724 0.124 0.000
17: 15 -2.639 0.229 0.000
18: 11 1 line 0 1
19: 12 1 line 2 3
20: 21 1 line 4 5
21: 22 1 line 6 7
22: 31 1 line 8 9
23: 32 1 line 10 11
24: 41 1 line 12 13
25: 42 1 line 14 15
26: 1 1
27: 交通密度データ, 台 /km
28: 11 33.7
29: 12 155.8
30: 21 52.4
31: 22 0.8
32: 31 3.2
33: 32 4.4
34: 41 185.2
35: 42 0.3
```

図4-6　ucd データ例

4.2.2 変換プログラムの作成

ここでは実例で行なった作業に沿って変換プログラムの作成について説明します。作成する変換プログラムは、ノード数やセル数などのヘッダ部分のデータ、ノードの座標値、セルを構成するノードの並び、セルデータを以下のようにして避難シミュレーションのデータから読み込みます。

- ノード座標

  ノードの座標については、以下の理由により避難シミュレーションのノードのx、y座標を変更します。避難シミュレーションでは同じ区間の道路を上下車線のリンクに分割した場合、お互いのリンクの両端のノードは同じノードを指しています。そのために、このままノードの座標をＡＶＳに読み込んで可視化した場合に上下車線のリンクが重なって表示されることになります。そこで、ノードの座標を上下車線方向にずらします。ノード番号は、新たな番号を付け直します。また、ucd データではノードは3次元上に定義しなければならないので避難シミュレーションの座標データにz座標の値を加えてucd データのノード座標とします。

- セル番号、セルを構成するノードの並び

  セル番号は、避難シミュレーションのリンク番号をそのまま使用します。セルタイプは、各リンクをlineセルで置き換えるためlineとします。セルを構成するノードはlineセルを構成する（新しく付け直した）ノード番号とします。

- セルデータ

  交通密度データをセルデータとして格納します。

ここでは、避難シミュレーションのデータの内容とそのフォーマットについて説明します。避難シミュレーションのデータは、シミュレーションを実行するために必要な入力データと実行した結果として得られる出力データから成ります。

- 入力データ

  道路網の構造を定義するデータとしてノードデータとネットワークデータがあります。

  1. ノードデータ（ファイル名 node.dat）

     隣接しているリンクとリンクの間の接点をノードと呼びます。ノードデータはノード番号（整数）及びノードの存在するx,y座標（実数）から成ります。ノードデータのファイルフォーマットは図4-7の通りです。

ノード番号	x座標	y座標
ノード番号	x座標	y座標
ノード番号	x座標	y座標
……	…	…

図4-7　ノードデータのファイルフォーマット

2. ネットワークデータ（ファイル名 link.dat）

リンクに関するデータです。ネットワークデータは、リンク番号（整数）、リンクに隣接しているノード番号（整数）などから成ります。ネットワークデータのファイルフォーマットは図 4-8の通りです。

リンク番号	発ノード	着ノード	…	…	…
リンク番号	発ノード	着ノード	…	…	…
リンク番号	発ノード	着ノード	…	…	…
……	……	……	…	…	…

図 4-8　ネットワークデータのファイルフォーマット

● 出力データ

1. リンクに関する出力データ（ファイル名 link.out）

避難シミュレーションは、１０秒毎にリンクに関するデータをファイル link.out に出力します。リンクに関する出力データは、シミュレーション時間（実数），リンク番号（整数）、交通密度データ（実数）、平均速度データ（実数）及び交通量データ（実数）から成ります。リンクに関する出力データのファイルフォーマットは図 4-9の通りです。

シミュレーション時間（０秒後）	リンク番号	交通密度データ	平均速度データ	交通量データ
シミュレーション時間（０秒後）	リンク番号	交通密度データ	平均速度データ	交通量データ
シミュレーション時間（０秒後）	リンク番号	交通密度データ	平均速度データ	交通量データ
……	……	……	……	……
シミュレーション時間（10 秒後）	リンク番号	交通密度データ	平均速度データ	交通量データ
シミュレーション時間（10 秒後）	リンク番号	交通密度データ	平均速度データ	交通量データ
シミュレーション時間（10 秒後）	リンク番号	交通密度データ	平均速度データ	交通量データ
……	……	……	……	……

図 4-9　リンクに関する出力データのファイルフォーマット

以上のことを考慮に入れて変換プログラムを作成します。変換プログラムは、必要なデータを避難シミュレーションの入出力データから読み込んで、ucd データのファイルフォーマットに従って変換します。

以下にC言語で作成した変換プログラムのソースファイル (create_data.c) を示します。

関数 set_header() でノード数、セル数等のヘッダ部分のデータ、node.dat 及び link.dat から得られるノード座標、セルを構成するノードの並びを header1.dat と header2.dat に保存しておきます。ノードの座標は関数 new_xyz() により上下車線の方向にずらした座標に変換します。

関数 set_cell() は、シミュレーションの経過時間毎にファイル (time0000.inp,time0001.inp,...) を生成して header1.dat 及び header2.dat に保存されているデータと link2.out から得られるセルデータを ucd データのファイルフォーマットに従って保存します。

```
/* 入力データ作成プログラム create_data.c */

#include <stdio.h>

#define NUM 1837 /* 総リンク数 */
#define COORD_Z 0.0 /* Z 座標 */
#define F_SIZE 30 /* ファイル名のサイズ */

void new_xyz(sx, sy, ex, ey);
void set_header();
void set_cell();

main()
{
  set_header();
  set_cell();
}
void set_header()
{
  FILE *in_link, *in_node, *ucd_h1, *ucd_h2;
  char buf[BUFSIZ];
  int link, start, end, node, num;
  float x, y, start_x, start_y, end_x, end_y;
  float z = COORD_Z;

  /* ネットワークデータをオープンする */
  in_link = fopen("link.dat", "r");

  /* header1.dat, header2.dat をオープンする */
  ucd_h1 = fopen("header1.dat", "w");
  ucd_h2 = fopen("header2.dat", "w");

  /* ノード数 セル数 データ数 を出力する */
  fprintf(ucd_h1, "%d %d 0 1 0\n", NUM*2, NUM);

  while(fgets(buf, BUFSIZ, in_link) != NULL){
    sscanf(buf, "%d%d%d", &link, &start, &end);
```

図4-10. ソースファイル (create_data.c) (その1)

```
      /* ノードデータをオープンする */
      if((in_node = fopen("node.dat", "r"))==NULL)
        fprintf(stderr, "can't open node.dat);
        exit(0);
      }

      while(fgets(buf, BUFSIZ, in_node) != NULL){
        sscanf(buf, "%d%f%f", &node, &x, &y);

        /* リンクに隣接するノードの座標を得る */
        if(node == start){
          start_x = x;
          start_y = y;
        }
        if(node == end){
          end_x = x;
          end_y = y;
        }

      }
      fclose(in_node);

      /* ノードの座標をずらす */
      new_xyz(&start_x, &start_y, &end_x, &end_y);

      /* header2.dat へセル番号及び接続ノードを出力する */
      fprintf(ucd_h2, "%4d 1 line %d %d\n", link, num, num+1);

      /* header1.dat へノード番号及びノード座標を出力する */
      fprintf(ucd_h1, "%4d %f %f %f\n", num++, start_x, start_y, z);
      fprintf(ucd_h1, "%4d %f %f %f\n", num++, end_x, end_y, z);

    }
    fclose(ucd_h1);
    fclose(ucd_h2);
    fclose(in_link);
}

void set_cell()
{
    FILE *out_link, *ucd_h1, *ucd_h2, *out;
    char buf[BUFSIZ], name[F_SIZE];
    int link, roop, frame = 0;
    float time, data;

    /* link2.out をオープンする */
    if((out_link = fopen("link2.out", "r")) == NULL)
      exit(0);

    while(fgets(buf, BUFSIZ, out_link) != NULL){
      sscanf(buf, "%f%d%f", &time, &link, &data);
```

図4-10. ソースファイル (create_data.c) (その2)

```
      if(frame < 10)
        sprintf(name, "ucd000%d.inp", frame);
      else if(frame >= 10 && frame < 100)
        sprintf(name, "ucd00%d.inp", frame);
      else if(frame >= 100 && frame < 1000)
        sprintf(name, "ucd0%d.inp", frame);
      else
        sprintf(name, "ucd%d.inp", frame);

      /* time*.inp をオープンする */
      out = fopen(name, "w");
      fprintf(stdout, "%s
      /* header1.dat, header2.dat ををオープンする */
      ucd_h1 = fopen("header1.dat", "r");
      ucd_h2 = fopen("header2.dat", "r");

      /* header1.dat, header2.dat の内容を time*.inp に出力する */
      while(fgets(buf, BUFSIZ, ucd_h1) != NULL)
        fputs(buf, out);
      while(fgets(buf, BUFSIZ, ucd_h2) != NULL)
        fputs(buf, out);

      fclose(ucd_h1);
      fclose(ucd_h2);

      /* リンク番号と交通密度データを出力する */
      fprintf(out, "%d %f \n", link, data);
      for(roop = 0; roop < NUM - 1; roop++){
        fgets(buf, BUFSIZ, out_link);
        sscanf(buf, "%f%d%f", &time, &link, &data);
        fprintf(out, "%d %f \n", link, data);
      }
      fclose(out);
      frame++;
      }
    fclose(out_link);
}
```

図 4-10. ソースファイル (create_data.c) (その3)

5　可視化プログラム（ＡＶＳネットワーク）の作成

本章では可視化プログラムの作成について解説します。可視化プログラムは数値データをコンピュータに読み込んで可視化のために必要な処理を行ない、その結果を画像として表示するプログラムです。ＡＶＳではモジュールを組み合わせて可視化プログラムを作成します。ＡＶＳによって作成した可視化プログラムをＡＶＳネットワーク（または、ネットワーク）といいます。

実例で作成するＡＶＳネットワークは、交通密度データを読み込んでそのデータの大きさに応じて色分けした画像を表示します。

ここでは、まずＡＶＳによるプログラミングの特長について解説します。次にＡＶＳネットワークを作成するために必要な準備について解説します。さらに実例に沿ってＡＶＳネットワークの作成及びモジュールの作成について説明します。（図5-1参照）

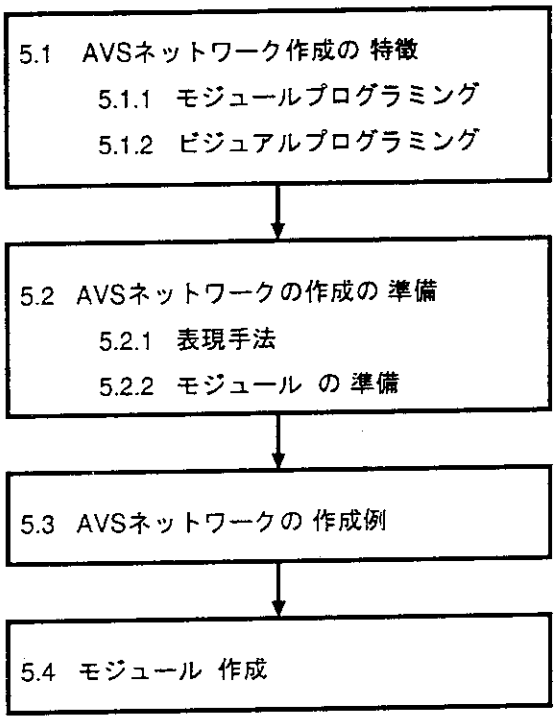

図 5-1　第５章の構成

5.1　ＡＶＳネットワーク作成の特徴

ＡＶＳによるプログラミングには、以下のような特長があります。

- モジュールプログラミング
  ＡＶＳでは、可視化処理を多段階のデータ変換処理を組み合わせたものと考えています。可視化処理の一部分を処理するプログラムをモジュールといい、可視化プログラム（ＡＶＳネットワーク）はモジュールを組み合わせて作成します。

- ビジュアルプログラミング
  ＡＶＳによるプログラミングでは、アイコンで表されたモジュールをマウスを使用して組み合わせます。ＡＶＳネットワークの作成は専用のエディタ（ネットワークエディタ）を使用して行ないます。

5.1.1 モジュールプログラミング

ここでは、特長の１つであるモジュールプログラミングについて説明します。モジュールは、その処理内容により４つのカテゴリに分類されます。どのタイプのデータを読み込んで、何をどのように表現し、表示するのかなどの処理の流れを考えてそのために必要なモジュールを選びだし、これを組み立ててＡＶＳネットワークを作成します。以下に４つのカテゴリとその処理内容を説明します。

- Data Input（データ入力モジュール）
  主にＡＶＳのフォーマットで記述したファイルを読み込むモジュール
  （例） read field, read ucd, read geometry, read image, read volume

- Filters（フィルタモジュール）
  主にデータの変換や抽出を行なうモジュール
  （例） field to ucd, ucd cell to node

- Mappers（マッパーモジュール）
  主に画像生成に必要な幾何データ（ジオメトリデータ）へ変換するモジュール
  （例） ucd counter, ucd cell color

- Data Output（データ出力モジュール）
  主に画像をディスプレイに表示するモジュール
  （例） geometry viewer, image viewer, graph viewer

ＡＶＳネットワークの作成例を図5-2に示します。モジュールは図のように長方形のアイコンで表されます。このようにＡＶＳでは、可視化の一部分の処理をするモジュールを組み合わせてプログラミングを行ないます。このようなプログラミングの方法をモジュールプログラミングといいます。

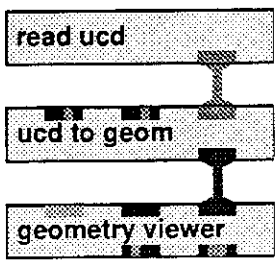

図 5-2　ＡＶＳネットワーク

5.1.2　ビジュアルプログラミング

ここでは、ビジュアルプログラミングについて説明します。ＡＶＳネットワークの作成では、ネットワークエディタ（図5-3参照）と呼ばれるＡＶＳ専用のエディタを使用します。ＡＶＳネットワークの作成は、マウスを使用して行ないます。モジュールパレットから必要な（長方形のアイコンで表示された）モジュールをワークスペースに移動し、それらのモジュールどうしを接続して可視化処理を組み立てます。ＡＶＳネットワークの作成は、ＣやＦＯＲＴＲＡＮなどのプログラミングと異なり、このようなＧＵＩ環境を利用してプログラミングを行ないます。このようなプログラミングをビジュアルプログラミングといいます。（[1] p6 参照）

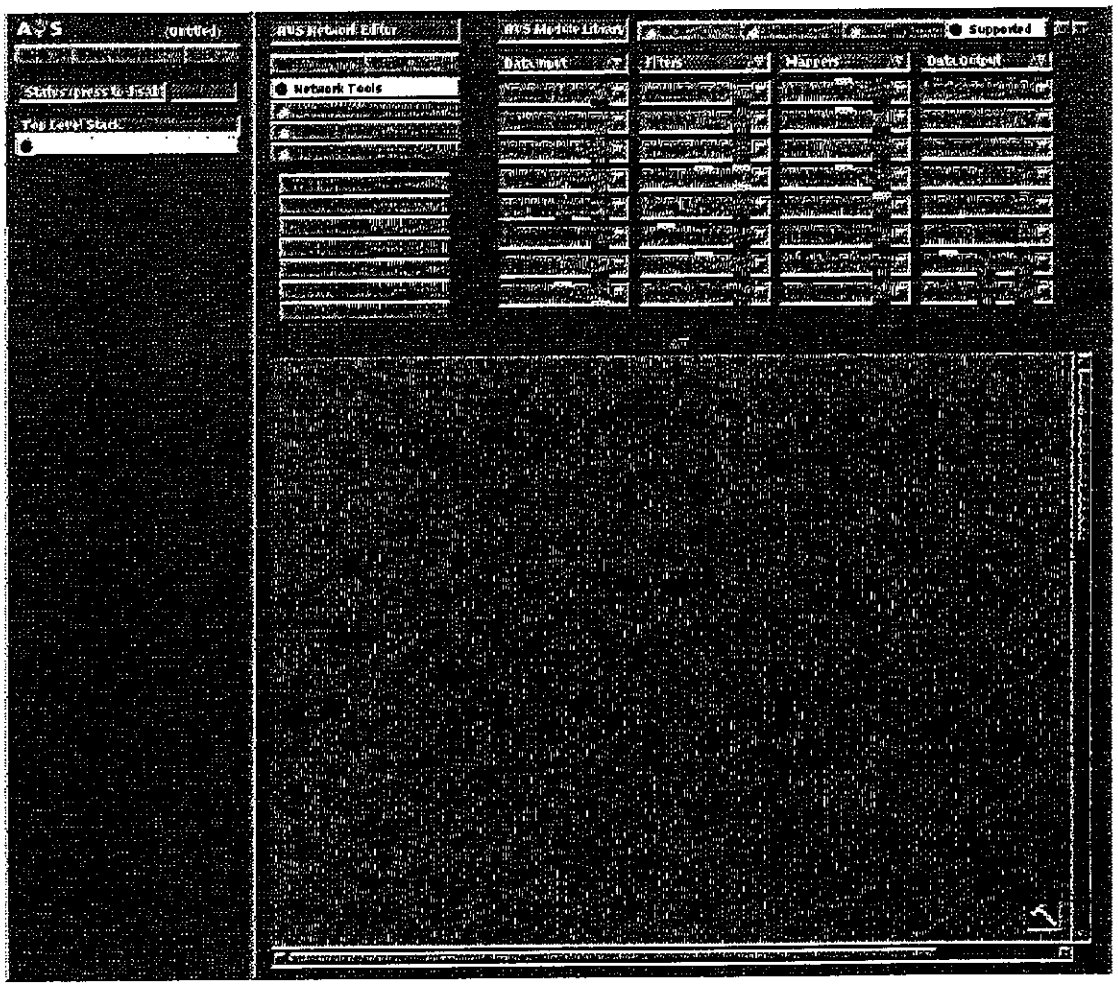

図 5-3　ネットワークエディタ

5.2 ＡＶＳネットワークの作成の準備

本節では、ＡＶＳネットワークを作成するために必要な準備について説明します。ＡＶＳネットワークを作成するには、まずデータの持つ性質を考慮してどのような手法で表現すれば分かりやすく可視化できるか（表現手法）を考えます。次にそれを実現するのに必要なモジュール（モジュールの準備）を決定します。

まず、表現手法について説明します。次に、モジュールの準備について説明します。

5.2.1 表現手法

シミュレーションにより得られたデータをどのような形式で表現するのかは、ＡＶＳはもとより第３者にも十分に分かりやすく可視化する上で、非常に重要な問題です。代表的な表現手法には、以下のようなものがあります。

- コンター図
  数値データをＲＧＢの色データに変換し、色の変化でデータの大きさを表します。
  コンター図ではデータの分布、傾向などを表現することができます。

- ベクトル図
  ベクトルデータを矢印などで表します。
  ベクトル図ではデータの方向と大きさを同時に表現することができます。

- 等値面
  同じ値を持つデータを結んで面で表します。
  等値面では同じ値を持つデータの分布などを表現することができます。

- グラフ
  データを折れ線グラフ、棒グラフで表します。
  グラフではデータの変化、傾向などを表現することができます。

- ボリュームレンダリング
  ボリュームデータから幾何データに変換せずに直接イメージデータに変換します。
  ボリュームレンダリングでは、全てのデータを同時に表示することができます。

実例では、避難シミュレーションの結果をコンター図で表現します。交通密度データの大きさに応じてルート（各リンク）を色分けして表現することで、どのルートに車両が集中しているのかを直観的に知ることができます。

5.2.2 モジュールの準備

ＡＶＳには標準で２００種類以上のモジュールが用意されています。モジュールは、ＡＶＳを起動してネットワークエディタを開くと、モジュールパレットに４つのカテゴリに分類されて表示されます。目的の可視化処理を実現するために必要なモジュールをモジュールパレットからネットワークエディタのワークスペースに移動します。カテゴリについては「5.1.1 モジュールプログラミング」を参照して下さい。

モジュールは入力ポート、出力ポート及びパラメータを持ちます。データは、入力ポートから読み込まれ、モジュールで変換して、出力ポートから出力されます。また、パラメータを変更することによって処理の条件を変えることができます。モジュールのポートの色は、そのポートから入出力できるデータのタイプを示しています。（[1] p16 参照）

パラメータはダイヤルやボタンなどのコントロールウィジェットにより変更することができます。また、パラメータポートを通して他のモジュールからパラメータ値を変更することもできます。

ＡＶＳの標準のモジュールに目的のモジュールがない場合には以下のような方法でモジュールを入手することができます。これらのモジュールは、ＡＶＳに取り込んで標準のモジュールと同様にして扱うことができます。

- PDS モジュール

  ＩＡＣ（International AVS Center）には、PDS モジュール（Public Domain Software module）が登録されています。ftp で以下のサイトに login して PDS モジュールを入手することができます。ＩＡＣには PDS モジュールのソースプログラムと Makefile を入手することができます。ソースプログラムを Makefile によりコンパイル、リンクして実行ファイルを作成してＡＶＳに読み込んで使用します。

  ftp.center.osaka-u.ac.jp（日本のサイト）

  以下のホームページに日本語で PDS モジュールの機能の説明が掲載されています。

  http://www/cssi.co.jp/hyper/KGT/seihin/AVS/avs.html

- サードパーティーから販売されているモジュール

  サードパーティーからいくつかのオプションのモジュールが販売されています。実例では、旭化成情報システム株式会社が販売しているＡＶＳビデオチェックキットを使用しています。

- モジュール作成

  目的のモジュールが、標準のモジュール、PDS モジュール、サードパーティー製モジュールなどにない場合は、C, ＦＯＲＴＲＡＮのプログラミング言語を使用して独自にモジュールを作成することができます。

  モジュールは Module Genetator モジュールを使用して骨格部分のソースプログラムを自動生成して、このソースプログラムに処理ルーチンを書き加えます。モジュールの作成については、「5.4 モジュール作成」で詳しく説明します。

5.3 ＡＶＳネットワークの作成例

ここでは、実例に沿ってＡＶＳネットワークの作成について説明します。各リンクを走行している車両数をコンター図で表現するＡＶＳネットワークを作成します。（[1] p15 参照）

まず、対象となる地域の道路網の概形を表示するＡＶＳネットワーク（その１）を作成します。（図 5-4参照）

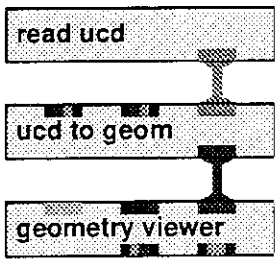

図 5-4　ＡＶＳネットワーク（その１）

ＡＶＳネットワーク（その１）では、read ucd モジュールを用いて第4章で作成した ucd データ（道路網に関するデータ及び各リンクの交通密度データ）を読み込みます。次に ucd to geom モジュールを用いて ucd データで定義された構造を幾何データ（geometry データ）に変換します。最後に geometry viewer モジュールを用いて幾何データからオブジェクトの生成、レンダリングを通して画像を表示します。

図 5-5にＡＶＳネットワーク（その１）を用いたときの表示例を示します。

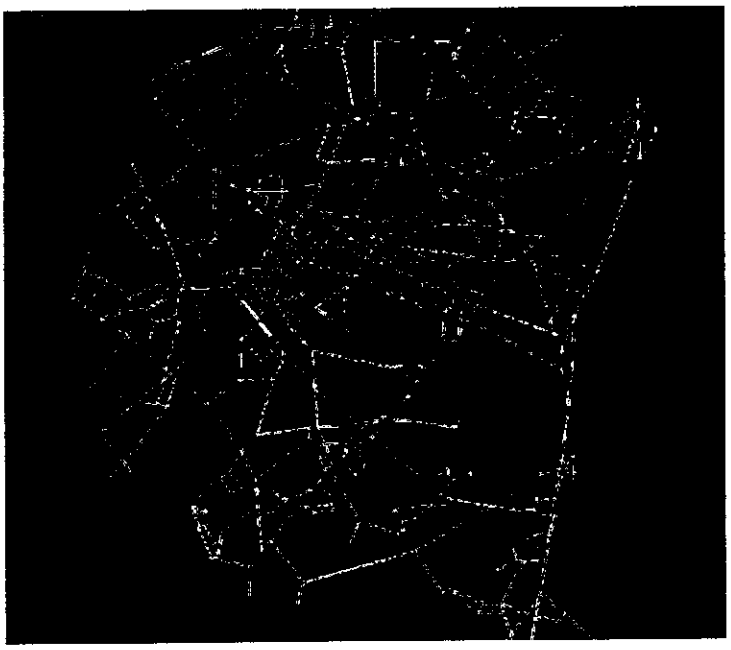

図 5-5　表示例（その１）

以下にＡＶＳネットワーク（その１）で使用しているモジュールについて説明します。

- read ucd(Data Input)
  入力ポート：文字列
  出力ポート：ucd データ
  ucd データのファイルフォーマットで記述されたデータを読み込むためのモジュールです。読み込むファイルを選択するためにはファイルブラウザを使用します。ファイルブラウザではディレクトリ名を赤字でファイル名を黒字で表示しています。read ucd モジュールではファイルブラウザに拡張子が.inp であるファイルのみを表示します。そのために ucd データのファイル名は ***.inp にしておく必要があります。ファイル名をクリックするとＡＶＳにその内容を読み込みます。

- ucd to geom(Mappers)
  入力ポート：ucd データ、field データ 1D 3-vector real（カラーデータ）
  出力ポート：geometry データ
  ucd データを幾何データ（geometry データ）に変換します。ucd データはモジュールの右側の入力ポートから読み込みます。実例では line セルを使用しているので ucd データによって与えられたノードの座標とノードの並びから、両端のノードを結んだ線分の集合を表す幾何データを生成します。

- geometry viewer(Data Output)
  入力ポート：geometry データ、
  field データ 2D/3D uniform byte scalar or 4-vector、colormap
  幾何データを読み込んでオブジェクトを生成してシーンを構築します。構築したシーンはファイルに保存しておくことができます。また、レンダリング（３次元のシーンを２次元の画像に変換すること）と画像の表示を行ないます。
  また、サブシステムとして以下のような操作をすることができます。

  - 幾何変換
    シーンを構成するオブジェクト、ライト、カメラ、テクスチャーに対する移動、拡大、縮小、回転などの幾何変換
  - オブジェクトに対する操作
    レンダリング方法の変更、反射属性の変更、テクスチャーマッピング
  - ライトに対する操作
    複数のライトの設定、各ライトのタイプの変更、ライトのカラーの設定
  - カメラに対する操作
    複数のカメラの設定、ビューボリュームの変更、レンダリング方法の変更

次に、交通密度の大きさに応じてセルを色分けして表示するためのＡＶＳネットワーク（その２）を作成します。（図 5-6参照）

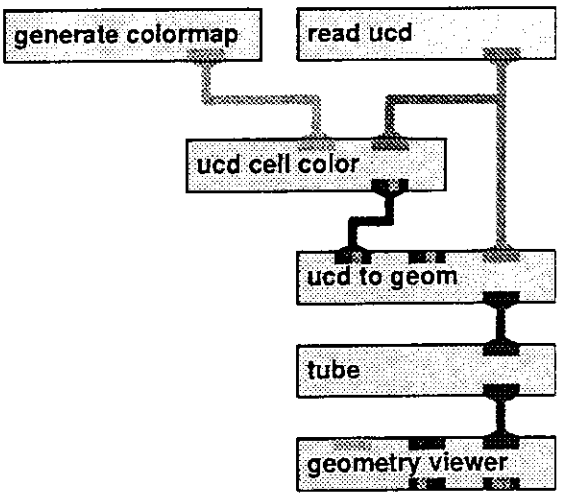

図 5-6　ＡＶＳネットワーク（その２）

ＡＶＳネットワーク（その２）では ucd cell color モジュールを用いて各セルデータの大きさに対応するカラーデータを生成します。generate colormap モジュールでは数値データとカラーデータの対応表を作成します。ucd to geom モジュールでは色の属性を持つ幾何データを生成します。また線を見やすく表示するために tube モジュールを用います。tube モジュールは線で表したオブジェクトをシリンダー状の管で表したオブジェクトに変換します。

ＡＶＳネットワーク（その２）を用いたときの表示例を図 5-7に示します。

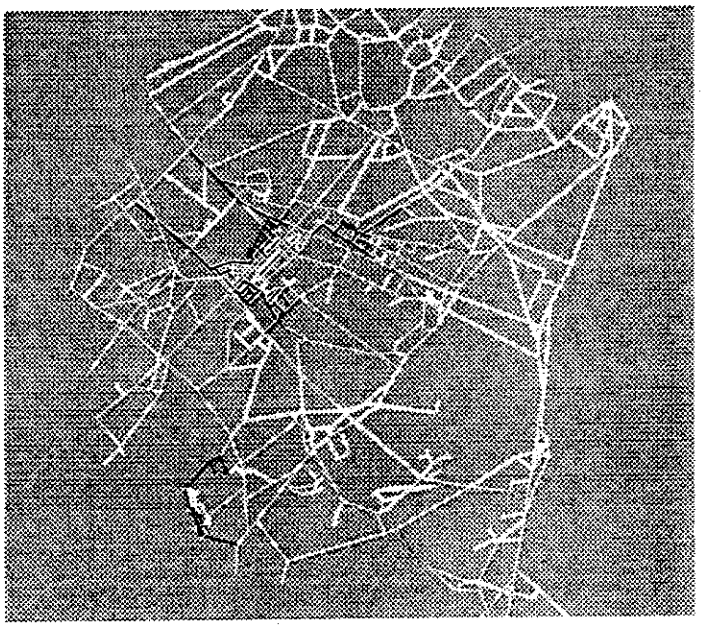

図 5-7　表示例（その２）

- generate colormap(Data Input)

  入力ポート：なし

  出力ポート：カラーデータ

  数値データとカラーデータの対応表（カラーマップ）を作成するモジュールです。入力データの値に応じたカラー値に変換するためにカラーデータを出力します。実例では、交通密度データとカラーデータの対応表を作成します。（表 5-1参照）

表 5-1　交通密度データとカラーデータの対応

0.0-0.5	白
0.5-28.0	水色
28.0-139.0	青
139.0-	赤

- ucd cell color(Mappers)

  入力ポート：ucd データ、カラーデータ

  出力ポート：field データ 1D 3-vector real

  ucd データ及びカラーデータを読み込み、セルデータをカラーデータに変換して出力します。

- ucd to geom(Mappers)

  入力ポート：ucd データ、field データ 1D 3-vector real

  出力ポート：geometry データ

  左の入力ポートから各セルに与えられたカラーデータを受け取ります。カラーデータは幾何データに加えられます。

- tube(Filters)

  入力ポート：geometry データ

  出力ポート：geometry データ

  line セルから生成した幾何データをシリンダー状の管で表した幾何データに変換します。パラメータ radius で管の半径を設定します。（図 5-8参照）

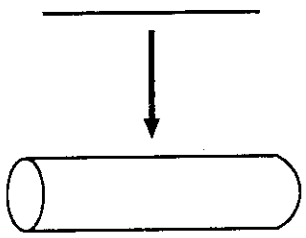

図 5-8　tube モジュール

5.4 モジュール作成

ここでは、モジュールの作成について説明します。ＡＶＳでは、適当な機能を持つモジュールが存在しない場合、C, ＦＯＲＴＲＡＮのプログラミング言語を使用して新しくモジュールを作成することができます。モジュールの作成は Module Generator モジュールでプログラムの骨格を生成した上で、そこに処理ルーチンを書き加えます。([1] p92 参照)

実例ではシミュレーションの経過時間を出力するモジュール (frame_to_time) を作成します。frame_to_time モジュールは、animated ucd モジュールが出力するフレーム数 (integer) を読み込んで、シミュレーションの経過時間を計算してその時間を文字列 (string) として出力します。animated ucd モジュールに関しては第 6章で説明します。なお、frame_to_time モジュールは第 5 章で作成するコマ撮り用のＡＶＳネットワークで使用します。

モジュール作成の作業手順は以下の通りです。

1. Module Generator モジュールを使用してプログラムの骨格を作成します。Module Generator モジュールでは、モジュール名、入力ポート、出力ポート、パラメータなどの設定を行ないます。ネットワークエディタを起動して Module Generator モジュールをワークスペースに移動すると以下の設定を行うためのコントロールウィジェットが表示されます。

   - モジュール名
     モジュール名を設定します。モジュール名はアイコンに表示される名前です。
     また、日本語も使用できます。
   - カテゴリ
     モジュールのカテゴリを設定します。
     モジュールの機能により Data Input,Filters,Mappers,Data Output に分類されます。
   - プログラム言語
     モジュール作成に使用するプログラム言語をＣ又は FORTRAN から選択します。
   - モジュールタイプ
     モジュールタイプを Subroutine モジュール,Coroutine モジュールから選択します。
   - 入力ポート、出力ポートの設定
     入力ポート、出力ポートのポート名及びデータタイプを設定します。
   - パラメータの設定
     パラメータ名、データタイプ及びコントロールウィジェットのタイプを設定します。
   - ソースファイル名
     ソースプログラムを保存するファイルの名前を設定します。

frame_to_time モジュールでは以下のように設定しました。

モジュール名	frame_to_time
モジュールのカテゴリ	Filters
プログラム言語	C
モジュールタイプ	Subroutine
入力ポート名	framenumber
入力ポートのデータタイプ	integer
出力ポート名	time
出力ポートのデータタイプ	string

2. ソースプログラムの骨格を生成します。Module Generator モジュールのファイルブラウザでファイル名を指定して、Write Source ボタンをクリックします。また、Include Hints ボタンをクリックした後で Write Source ボタンをクリックすると、処理ルーチン作成のためのヒントも同時に保存できます。

3. vi や emacs などのエディタを利用して、ソースプログラムの骨格に処理ルーチンを書き加えます。実例では、プログラムに読み込んだフレーム数からシミュレーション経過時間を算出する処理及び経過時間を文字列として出力する処理を記述します。

4. Write Makefile ボタンをクリックすることにより Makefile を自動生成します。
   ファイル名は Makefile. モジュール名で生成されます。（例 Makefile.frame_to_time）

5. Makefile を用いて実行ファイルを作成します。
   % make -f Makefile 名

6. 実行ファイルをＡＶＳに読み込みます。ネットワークエディタの Module Tools ボタンをクリックして、さらに Read Module(s) ボタンをクリックするとファイルブラウザが表示されます。実行ファイルのディレクトリに移動して、実行ファイル名をクリックするとＡＶＳにモジュールを読み込むことができます。

以下に frame_to_time モジュールのソースプログラム（frame_to_time.c）を示します。

```
#include <stdio.h>
#include <avs/avs.h>
#include <avs/port.h>

/* Module Description */
int frame_to_time_desc()
{

  int in_port, out_port, param, iresult;

  extern int frame_to_time_compute();
```

図5-9. ソースファイル（frame_to_time.c）（その１）

```
  AVSset_module_name("frame_to_time", MODULE_FILTER);

  /* Input Port Specifications */
  in_port = AVScreate_input_port("framenumber", "integer", REQUIRED);

  /* Output Port Specifications */
  out_port = AVScreate_output_port("time", "string");

  AVSset_compute_proc(frame_to_time_compute);

  return(1);
}

/* 処理ルーチン */
/* この部分をユーザが作成する */

/* ここから */
int frame_to_time_compute( framenumber, time)
int framenumber;
char **time;
{
  int sec, hour, second;
  sec = framenumber/6;
  hour = sec/60;
  second = sec%60;

  *time = (char *)malloc(BUFSIZ*sizeof(char));

  if(second<10)
    sprintf(*time, "%d:0%d", hour, second);
  else
    sprintf(*time, "%d:%d", hour, second);

  return(1);
}
/* ここまで */

/* Initialization for modules contained in this file.  */
static int ((*mod_list[])()) = {
  frame_to_time_desc
};
#define NMODS (sizeof(mod_list) / sizeof(char *))

AVSinit_modules()
{
AVSinit_from_module_list(mod_list, NMODS);
}
```

図 5-9. ソースファイル (frame_to_time.c) （その２）

6　画像の保存

本章では、可視化した結果の保存について解説します。ＡＶＳでは可視化により得られた画像をファイルにセーブしたり、プリンタで印刷したり、ビデオテープに録画したりして保存することができます。

実例では、可視化した結果をビデオテープに保存します。実例で作成したＡＶＳネットワーク（図 5-6）は任意の時刻の避難状況を可視化して画像を表示します。避難シミュレーションが１０秒毎に出力する時系列データを経過時間の順番に可視化して、その画像をコマ撮りしていくことによりアニメーションとして保存することができます。

ここでは、まず、アニメーション作成の概要について解説します。次に、本作業で使用する動画像処理システム及びＡＶＳビデオチェックキットについて解説します。最後に、実例に沿ってアニメーションの作成について解説します。（図 6-1参照）

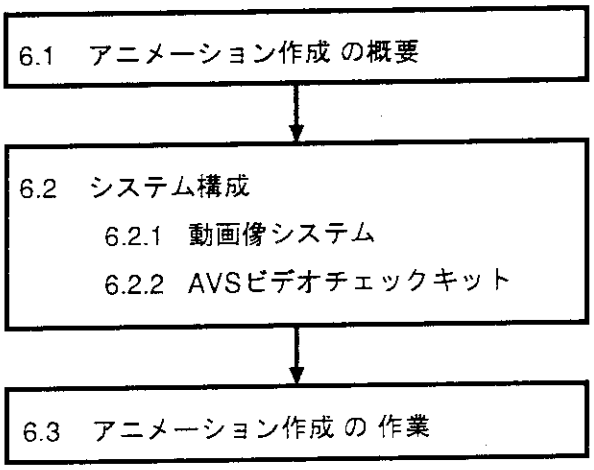

図 6-1　第６章の構成

6.1 アニメーション作成の概要

実例では以下のようにしてアニメーションを作成しています。（図6-2参照）

1. 避難シミュレーションのデータをucdデータに変換してファイルに保存します。避難シミュレーションは、各リンクの交通密度データを１０秒毎に出力します。実例では、ucdデータに変換されたデータを以下のような名前のファイルに保存しています。ファイル名"time"の後に続く数字がデータを読み込む順番を示します。（第4章 入力データの作成を参照）

0秒後のデータ	time0000.inp
10秒後のデータ	time0001.inp
20秒後のデータ	time0002.inp
30秒後のデータ	time0003.inp
40秒後のデータ	time0004.inp
50秒後のデータ	time0005.inp
...	.....

2. ucdデータをファイル名で示している順番でＡＶＳネットワークに読み込みます。

3. ＡＶＳネットワークはucdデータに変換されたデータを可視化して画像を表示します。

4. 画像を指定したコマ数分ビデオテープに録画します。

以下、２～４の作業を繰り返してアニメーションを作成します。実例ではこの２～４の作業をＡＶＳビデオチェックキットを使用して自動的に行なっています。

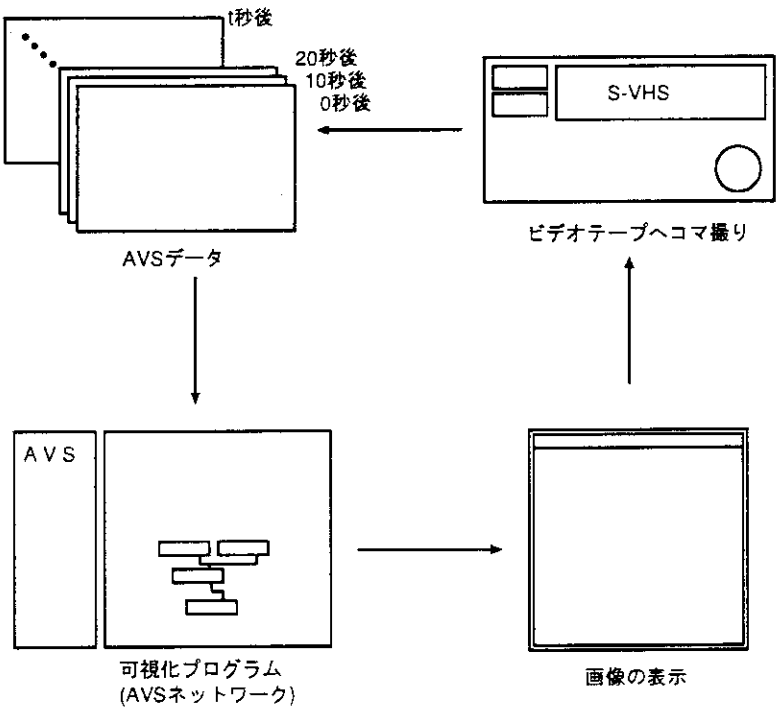

図 6-2　アニメーションの作成

6.2 システム構成

実例では、アニメーションを作成するにあたって情報システム管理課で管理している動画像処理システムを使用しています。また、コマ撮りを自動的化するためにファイルの連続読み込みやコマ撮り用コントローラの制御などの機能を持つＡＶＳビデオチェックキットも使用しています。ＡＶＳビデオチェックキットは、ＡＶＳのモジュールとして提供されます。ただし、ＡＶＳビデオキットを使用するには、ＡＶＳの Version5.0 以上（5.0, 5.01, 5.02, 5.3）が必要です。

6.2.1 動画像処理システム

動画像処理システムのシステム構成は以下の通りです。（図 6–3参照）

- ワークステーション
  HP9000/730
  （ＡＶＳ Version5.02 及びＡＶＳビデオチェックキットが導入されています）
- コマ撮り対応のＶＴＲ（以下 ＶＴＲ という）
  Victor BR-S822B
- モニタ
  Victor VM-R150S
- フレームスキャンコンバータ（以下 ＦＳＣ という）
  （株）フォトロン FSC-64000vz
- コマ撮り用コントローラ
  （株）フォトロン RS-422A

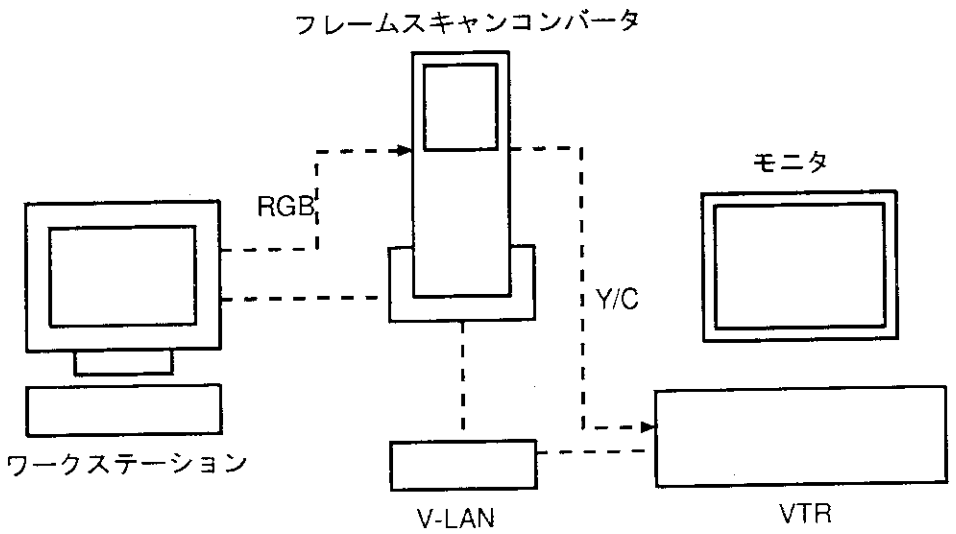

図 6–3　動画像処理システム

6.2.2 ＡＶＳビデオチェックキット

ここでは、ＡＶＳビデオチェックキットについて説明します。ＡＶＳビデオチェックキットは、コマ撮りを自動的に行なうためのＡＶＳのモジュール群です。ＡＶＳビデオチェックキットは主にファイルを連続して読み込むために使用するコルーチンモジュール及びコマ撮り用コントローラを制御するモジュールから成ります。以下にそれぞれのモジュールについて説明します。

- コルーチンモジュール

  ＡＶＳビデオチェックキット附属のコルーチンモジュールを使って連続して表示させるためには（time0000.inp,time0001.inp, ... ）のようなファイル名をつけておく必要があります。ＡＶＳビデオチェックキットには、以下のようなコルーチンモジュールがあります。最初に読み込むファイル名を指定して sleep ボタンをクリックすれば、連続的にファイルを読み込むことができます。

  - animated ucd
    ucd データ読み込み用のコルーチンモジュールです。
  - animated field
    field データ読み込み用のコルーチンモジュールです。
  - animated geom
    geometry データ読み込み用のコルーチンモジュールです。
  - animated image
    image データ読み込み用のコルーチンモジュールです。

  また、これらのコルーチンモジュールはファイル名の読み込み順を表す数字（integer）の部分を文字列として出力します。実例では、この数字から frame_to_time モジュールを用いてシミュレーションの経過時間を求めています。

- コマ撮り用コントローラの制御モジュール

  ＶＴＲはコマ撮り用コントローラによって制御します。さらにコマ撮り用コントローラモジュールによってコマ撮り用コントローラを制御します。コマ撮り用コントローラの制御モジュールは、録画開始時間、録画フレーム数をパラメータで与えるだけでコマ撮りを自動的に行います。実例ではコマ撮り用コントローラに V_LAN を使用しています。V_LAN の制御用モジュールとして v_lan controller モジュールを使用します。v_lan controller モジュールの使用方法については、参考文献 [1] p126 を参照して下さい。

6.3 アニメーション作成の作業

1. コマ撮り用のモジュールの読み込み

コマ撮り用のモジュールをＡＶＳに読み込みます。ここでは animated ucd モジュールをＡＶＳに読み込む場合についてその方法を説明します。（[1] p31 参照）

この方法は、自作したモジュールをＡＶＳに読み込む場合にも有効です。

(a) ネットワークエディタを起動します。

(b) ネットワークエディタの Module Tools ボタンをクリックして Module Tools メニューを開きます。

(c) Read Module(s) ボタンをクリックするとファイルブラウザが表示されます。

(d) モジュールがインストールされているディレクトリに移動します。（[1] p10 参照）
animated ucd モジュールは /usr/avs/vtr/tools/animated_ucd にあります。

(e) ファイルブラウザに表示されているモジュールの実行ファイル名（ucd_anim ）をクリックすると、そのモジュールを読み込んでモジュールパレットにモジュール名（animated ucd）が表示されます。

2. コマ撮り用のＡＶＳネットワークの作成

自動的にコマ撮りを行なうＡＶＳネットワークを作成します。（図 6–4参照）コマ撮り用のＡＶＳネットワークの作成には、以下のモジュールが必要です。

- 連続してデータを読み込むモジュール（animated ucd モジュール）
- ＡＶＳと同期をとり画像をコマ撮りするモジュール（v_lan controller モジュール）

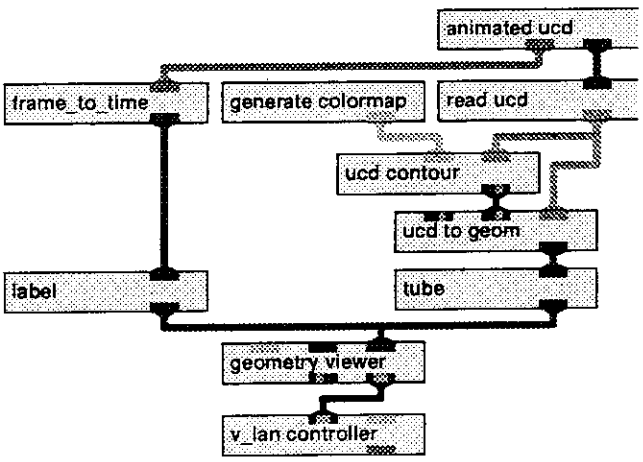

図 6–4　コマ撮り用ＡＶＳネットワーク

実例では animated ucd モジュールと v_lan controller モジュールを用いて図 6-4のようなコマ撮り用ＡＶＳネットワークを作成します。コマ撮り用ＡＶＳネットワークでは、現在表示されている画像をパラメータで設定したコマ数分録画するために v_lan controller モジュールを用いてＶＴＲを制御します。v_lan controller モジュールは録画を終了して待機状態になると、animated ucd モジュールに対して終了のフラグを送ります。animated ucd モジュールはフラグを受けると、次のファイルを読み込み再び画像を表示します。v_lan controller モジュールの実行ファイルは /usr/avs/vtr/vtr_cont/v_lan にあります。

3. タイムコードの記録

コマ撮り用のＶＴＲは、ビデオテープに記録されたタイムコードによりコマ撮りを制御しています。このタイムコードはコマ撮りを行う前にビデオテープに記録しておく必要があります。タイムコードはフレーム単位で記録されます。なお、１秒間は３０フレームです。コマ撮り用のＶＴＲは、このタイムコードを元に画像をフレーム単位で記録しています。（[1] p114 参照）

4. コマ撮りに必要な機器の準備

コマ撮りに必要な機器の準備を行ないます。実例では、ＶＴＲ、ＦＳＣ、コマ撮り用コントローラを使用します。ＶＴＲにビデオテープを入れて各スイッチをコマ撮り用にセットしておきます。また、モニタを見ながらＦＳＣを操作して録画したい画像の位置や大きさを調整します。さらに、録画中にスクリーンセーバが起動しないようにしておくことも重要です。（[1] p118, p119, p124 参照）

5. パラメータの設定

各モジュールのパラメータを設定します。animated ucd モジュールでは、最初に読み込むファイルを選択します。なお、連続データは同じディレクトリに置いておく必要があります。また、v_lan controller モジュールの録画開始時間、画像１枚に当たりの録画フレーム数などのパラメータを設定します。（[1] p126 参照）

6. コマ撮りの実行

これまでに説明したコマ撮りの準備が完了したら、以下のようにしてコマ撮りを実行します。

(a) v_lan controller モジュールの Auto Record ボタンをクリックします。現在表示されている画像が録画されます。

(b) コルーチンモジュールの sleep ボタンをクリックするとＡＶＳネットワークは番号順にファイルを読み込んでコマ撮りを行ないます。また、animated ucd モジュールはフレーム数を出力します。作成したコマ撮り用のＡＶＳネットワーク（図 6-4）では、frame_to_time モジュールによりフレーム数から現在のシミュレーション時間を出力します。

(c) 最後のファイルを読み込んで画像を録画するとコマ撮りの終了です。終了するとＶＴＲは待機状態となり、v_lan controller モジュールの Stop ボタンをクリックしてＶＴＲを停止します。

7　おわりに

本報告書では「避難シミュレーション」を題材としてシミュレーション結果をＡＶＳに読み込んで可視化し、画像をＶＴＲに記録するまでの作業について解説した。ユーザは、本報告書に従って上記の題材をＡＶＳで処理していくことでＡＶＳ利用に必要な一連の作業手順の修得が可能である。今後はより高度なＡＶＳ利用について利用手引を整備したいと考えている。

謝辞

本報告書の作成する上で、「避難シミュレーション結果の可視化」に関する資料を緊急時システム研究室の石神努氏、梅本道孝氏、 田中幹人氏に提供していただきました。また、本報告書の内容について数値実験技術開発グループの渡辺正氏、外来研究員の田辺豪信氏に検討していただきました。ここに感謝の意を表します。

最後に本報告書を執筆する機会を与えて下さいました計算科学技術推進センター長浅井清氏、次長秋元正幸氏（現原子炉安全工学部長）に感謝致します。

参考文献

[1] 増子 献児, 五来 一夫, 加藤 克海, 山崎 和彦, ＡＶＳ利用手引書, JAERI-Tech 95-008(1996).

[2] 加藤 克海, 渡辺 正, 町田 昌彦, 蕪木 英雄, 動画像処理システムの開発, JAERI-Tech 95-014(1995).

[3] 加藤 克海, 渡辺 正, 蕪木 英雄, リアルタイムモニターシステムの開発, JAERI-Tech 96-044(1996).

[4] T.Ishigami, M.Tomizawa and K.Kobayashi, "Transportation Evacution Simulation with STREEM, "29th ISATA,Dedicated Conf. on Advanced Transportation Telematics/ITS-A New Era of Mobility and Safety,349(1996).

[5] クボタコンピュータ株式会社, AVS USER'S GUIDE(1991).

[6] クボタコンピュータ株式会社, AVS MODULE REFERENCE(1993).

[7] クボタコンピュータ株式会社, ＡＶＳハンドブック (1993).

[8] クボタコンピュータ株式会社, ＡＶＳハンドブック Vol.2(1994).

[9] オーム社, AVS による実践ビジュアライゼーション (1995).

[10] 日本ビクター株式会社, ビデオカセットレコーダー BR-S822B 取扱説明書.

[11] 日本ビクター株式会社, カラービデオモニター VM-R150S 取扱説明書.

7　おわりに

本報告書では「避難シミュレーション」を題材としてシミュレーション結果をＡＶＳに読み込んで可視化し、画像をＶＴＲに記録するまでの作業について解説した。ユーザは、本報告書に従って上記の題材をＡＶＳで処理していくことでＡＶＳ利用に必要な一連の作業手順の修得が可能である。今後はより高度なＡＶＳ利用について利用手引を整備したいと考えている。

謝辞

本報告書の作成する上で、「避難シミュレーション結果の可視化」に関する資料を緊急時システム研究室の石神努氏、梅本道孝氏、田中幹人氏に提供していただきました。また、本報告書の内容について数値実験技術開発グループの渡辺正氏、外来研究員の田辺豪信氏に検討していただきました。ここに感謝の意を表します。

最後に本報告書を執筆する機会を与えて下さいました計算科学技術推進センター長浅井清氏、次長秋元正幸氏（現原子炉安全工学部長）に感謝致します。

参考文献

[1] 増子 献児, 五来 一夫, 加藤 克海, 山崎 和彦, ＡＶＳ利用手引書, JAERI-Tech 95-008(1996).

[2] 加藤 克海, 渡辺 正, 町田 昌彦, 蕪木 英雄, 動画像処理システムの開発, JAERI-Tech 95-014(1995).

[3] 加藤 克海, 渡辺 正, 蕪木 英雄, リアルタイムモニターシステムの開発, JAERI-Tech 96-044(1996).

[4] T.Ishigami, M.Tomizawa and K.Kobayashi, "Transportation Evacution Simulation with STREEM, "29th ISATA,Dedicated Conf. on Advanced Transportation Telematics/ITS-A New Era of Mobility and Safety,349(1996).

[5] クボタコンピュータ株式会社, AVS USER'S GUIDE(1991).

[6] クボタコンピュータ株式会社, AVS MODULE REFERENCE(1993).

[7] クボタコンピュータ株式会社, ＡＶＳハンドブック (1993).

[8] クボタコンピュータ株式会社, ＡＶＳハンドブック Vol.2(1994).

[9] オーム社, AVS による実践ビジュアライゼーション (1995).

[10] 日本ビクター株式会社, ビデオカセットレコーダー BR-S822B 取扱説明書.

[11] 日本ビクター株式会社, カラービデオモニター VM-R150S 取扱説明書.

7　おわりに

本報告書では「避難シミュレーション」を題材としてシミュレーション結果をＡＶＳに読み込んで可視化し、画像をＶＴＲに記録するまでの作業について解説した。ユーザは、本報告書に従って上記の題材をＡＶＳで処理していくことでＡＶＳ利用に必要な一連の作業手順の修得が可能である。今後はより高度なＡＶＳ利用について利用手引を整備したいと考えている。

謝辞

本報告書の作成する上で、「避難シミュレーション結果の可視化」に関する資料を緊急時システム研究室の石神努氏、梅本道孝氏、 田中幹人氏に提供していただきました。また、本報告書の内容について数値実験技術開発グループの渡辺正氏、外来研究員の田辺豪信氏に検討していただきました。ここに感謝の意を表します。

最後に本報告書を執筆する機会を与えて下さいました計算科学技術推進センター長浅井清氏、次長秋元正幸氏（現原子炉安全工学部長）に感謝致します。

参考文献

[1] 増子 献児, 五来 一夫, 加藤 克海, 山崎 和彦, ＡＶＳ利用手引書, JAERI-Tech 95-008(1996).

[2] 加藤 克海, 渡辺 正, 町田 昌彦, 蕪木 英雄, 動画像処理システムの開発, JAERI-Tech 95-014(1995).

[3] 加藤 克海, 渡辺 正, 蕪木 英雄, リアルタイムモニターシステムの開発, JAERI-Tech 96-044(1996).

[4] T.Ishigami, M.Tomizawa and K.Kobayashi, "Transportation Evacution Simulation with STREEM, "29th ISATA,Dedicated Conf. on Advanced Transportation Telematics/ITS-A New Era of Mobility and Safety,349(1996).

[5] クボタコンピュータ株式会社, AVS USER'S GUIDE(1991).

[6] クボタコンピュータ株式会社, AVS MODULE REFERENCE(1993).

[7] クボタコンピュータ株式会社, ＡＶＳハンドブック (1993).

[8] クボタコンピュータ株式会社, ＡＶＳハンドブック Vol.2(1994).

[9] オーム社, AVS による実践ビジュアライゼーション (1995).

[10] 日本ビクター株式会社, ビデオカセットレコーダー BR-S822B 取扱説明書.

[11] 日本ビクター株式会社, カラービデオモニター VM-R150S 取扱説明書.