移動体の駆動をシミュレートする ファジィエキスパートシステム

広 内 哲 夫

Fuzzy Expert Systems Simulating Drive Control of Vehicles

Tetsuo Hirouchi

The best way for beginners to learn fuzzy theory is to make use of simple fuzzy expert systems which fuzzy inference is applied to. In this paper, we explain two fuzzy expert systems for the beginners. One is the simulator for a missile horming a target, another is that for a car automatically cotrolling its own position and velocty. The fuzzy inference engines have several types of simple membership functions, and they can calculate exact position, velocity and etc.

1 はじめに

現在，産業の分野，特に制御の分野で，ファジィ理論の応用が盛んに進められ，多くの成果が生み出されている。一方，心理学におけるアンケート調査の際の一般の人からの意見の収集，あるいは経営管理におけるエキスパートシステム構築の際の管理者からの経験やノウハウの習得において，曖昧さの処理が含まれることから，産業分野以外でもファジィ理論の応用が期待されている。

ファジィ理論では，メンバーシップ関数の設定が微妙であり，その取り扱いによっては，結果に大きな影響が出る。従って，理論の適用に当たっては，その点を充分確認しておく必要がある。このような観点から，筆者はファジィ理論の特性や有効性を初心者が視覚的に認識することが出来るファジィエキスパートシステムを開発したので報告する。本論文では，目標物体を正確に追尾するミサイル，および位置・速度・停止を制御する自動車のシミュレーションをそのテーマにしている。これらのシミュレーションはファジィ理論に基づくファジィ制御の応用であり，制御のための規則は専門家（エキスパート）の知識に依っている。その結果はコンピュータ・グラフィックスによる動的な図形として描き出される。

2 ファジィ制御の方法

2.1 ファジィ集合論

ファジィ制御を説明するまえに，その基礎となるファジィ集合について簡単に述べておく。フ

ァジィ集合の概念は，1965年，カリフォルニア大学の Zadeh によって提案された。これまでの集合論では，要素が集合に属するか属さないかの評価を2値評価で行うが，ファジィ集合論はその評価を無限値評価で行えるように拡張したものである。

いま全体集合 X の中で，境界の曖昧な部分領域 A を考える。これをファジィ集合と呼び，メンバーシップ関数 $m_A(x)$ を用いて，次のように定義される。

$$m_A : X \rightarrow [0, 1]$$

メンバーシップ関数 $m_A(x)$ は，全体集合 X の要素 x がファジィ集合 A に属すると思われる度合い（グレード）を表わしており，この度合いは所属度と呼ばれる。ファジィ集合 A はメンバーシップ関数を用いると，次のように表現される。

$$A = m_A(x_1)/x_1 + m_A(x_2)/x_2 + \cdots\cdots + m_A(x_n)/x_n$$
$$= \sum_{i=1}^{n} m_A(x_i)/x_i$$

ここで，／は分離記号，＋は OR 結合を意味する。この式の用い方の具体例としては，例えば1年から10年先までの10年間を全体集合 X とすると，その間で“近い将来”というファジィ集合 A は

$$A = 0.2/1\text{年先} + 0.6/2\text{年先} + 1.0/3\text{年先} + 0.7/4\text{年先} + 0.1/5\text{年先}$$

で表すことが出来る。ここで0.7/4は，$m_A(4)$の値が0.7であることを示している。これは4年先が近い将来に属する度合が0.7であることを意味している。なお，メンバーシップ関数の値が0の場合は，その記述を省略する。

ファジィ集合論では，メンバーシップ関数は，主観的評価によって自由に定めてよい。後述するファジィ制御においては，全体集合 X が連続的な数値を取ることから，そのファジィ集合 A を定義するメンバーシップ関数は直線の組合せで表現される。多くの場合には，つぎの式で表される三角型が用いられる。なお，∨の演算は min 演算であり，2つの値から小さい方の値をとる演算である。

$$A(x) = (1/a)(-|x-b| + a) \vee 0 \quad (a > 0)$$

全体集合 X 上のファジィ集合 A，B における集合演算は，メンバーシップ関数を用いて次のように定義される。ただし，x は全体集合 X に属する要素である。

包　含：$A \subseteq B \longleftrightarrow m_A(x) \leqq m_B(x)$
積集合：$A \cap B \longleftrightarrow m_{A \cap B}(x) = m_A(x) \wedge m_B(x)$
和集合：$A \cup B \longleftrightarrow m_{A \cup B}(x) = m_A(x) \vee m_B(x)$
補集合：$\sim A \longleftrightarrow m_{\sim A}(x) = 1 - m_A(x)$

ここで，∧，∨は，min 演算，max 演算であり，2つの値からそれぞれ小さい方の値，大きい方の値をとる演算である。

2.2 ファジィ推論

我々の日常的な推論は，常に曖昧さを伴っており，このような推論はファジィ推論と呼ばれる。本節では，ファジィ推論の方法について述べる。

無限値論理であるファジィ論理におけるファジィ命題は一般に

「x is A」

という形式で表され，x は主語，A は述語であり，A はファジィ集合によってその曖昧さが表現される。ファジィ推論の応用において重要な命題は，この要素命題から構成される複合命題であ

る。これは

$$「x \text{ is } A」 \rightarrow 「y \text{ is } B」$$

で表される“含意”と呼ばれるファジィ条件命題である。これは

$$\text{if } 「x \text{ is } A」 \text{ then } 「y \text{ is } B」$$

という意味（規則）を表しており，ファジィ制御においては，これをファジィプロダクション規則と呼ぶ。

ファジィ推論の基本の一つは，2値論理である記号論理における modus ponens の一般化である。modus ponens とは，「P ならば Q が真」であるとき，「P が真」ならば「Q は真」であることを推論することで，以下のように表される。

前提1	$P \rightarrow Q$
前提2	P
結論	Q

一般化された modus ponens，すなわち fuzzy modus ponens は次のように定められる。A，A' を全体集合 X 上のファジィ集合，B，B' を全体集合 Y 上のファジィ集合とし，A と A'，および B と B' はそれぞれ必ずしも一致していないものとすると，次のようなファジィ集合を用いた曖昧な前提1と前提2から，同じく曖昧な結論を導きだそうとするものである。

前提1	$「x \text{ is } A」 \rightarrow 「y \text{ is } B」$
前提2	$「x \text{ is } A'」$
結論	$「y \text{ is } B'」$

ところで，上記の前提1のファジィ条件命題は，ファジィ関係 R を用いて，主語を (x, y) とし，述語を R とする一つの命題

$$「(x, y) \text{ is } R」$$

に変換することが出来る。これは A と B との間の何等かのファジィ的な因果関係を表していると考えられる。ファジィ関係 R は形式的に

$$R = A \rightarrow B$$

と表される。ファジィ関係 R は，全体集合 X，Y との直積 $X \times Y$ 上のファジィ集合であると考えられる。従って，fuzzy modus ponens による推論とは，全体集合 X 上のファジィ集合 A' とファジィ関係 R との間で次の合成

$$B' = A' \circ R$$

を行うことにより，全体集合 Y 上のファジィ集合 B' を求めることである。これをファジィ推論の合成規則と呼ぶ。なお，演算記号 $\circ$ は max-min 合成演算を示す。

ファジィ関係 R はファジィ集合 A，B を用いて構成されるが，その構成方法は唯一ではない。そこで，ここでは，Mamdani の用いた次の方法

$$R = A \times B$$

すなわち，A と B の直積を採用することにし，max-min 合成演算を適用すると，推論結果から得られるファジィ集合 B' のメンバーシップ関数 $m_{B'}(y)$ は次の通りになる。ただし，$m_A(x)$，$m_{A'}(x)$，$m_B(y)$ はそれぞれファジィ集合 A，A'，B のメンバーシップ関数である。

$$m_{B'}(y) = a \wedge m_B(y)$$

ここで，a は次の式で示される。

$$a = \bigvee_x [m_A(x) \wedge m_{A'}(x)]$$

上式の解釈は次のように行われる。a はファジィ集合 A，A' がどの程度重なり合っているかを示す尺度と考えられるので，この尺度を適合度と呼ぶと，推論結果のファジィ集合 B' のメンバーシップ関数 $m_{B'}(y)$ は，図1に示すように，この適合度数 a で，ファジィ集合 B のメンバーシップ関数 $m_B(y)$ の頭の部分をカットしたものと考えることが出来る。そのため，この方法は頭切り法と呼ばれる。

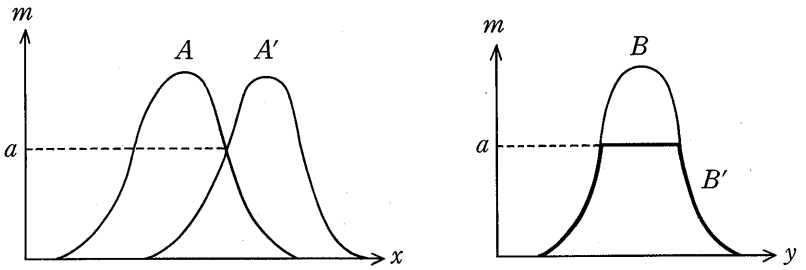


図1　頭切り法による推論

2.3　ファジィ制御

ファジィ推論は制御の分野にも応用され，これをファジィ制御と呼ぶ。前述したファジィ推論をファジィ制御にも適用できるように拡張してみる。Zadeh は制御のアルゴリズムを if-then 規則，すなわちファジィプロダクション規則で記述することを提唱した。システムの状態を表す変数（後述のファジィ推論エンジンの立場からすると入力変数）を x，y，システムへの入力に関する変数（同じく後述のファジィ推論エンジンの立場からすると出力変数）を z とすると，システムを制御する n 個からなるファジィプロダクション規則は次のように表わされる。

$$\text{if}\quad x_i=A_i\quad\text{and}\quad y_i=B_i\quad\text{then}\quad z_i=C_i\qquad(i=1,\ 2,\ \cdots,\ n)$$

ここで，A_i，B_i，C_i は全体集合 X，Y，Z 上のファジィ集合であり，変数 x_i，y_i，z_i のとるファジィ値である。これはファジィ変数とも呼ばれ，この集合の名前が言語ラベルとなる。if と then の間の命題は前件部，then の後の命題は後件部と呼ばれる。ファジィプロダクション規則は，例えば次のように用いられる。

if　速度＝遅い　and　加速度＝ほぼゼロ　then　アクセル＝踏み込め

if　速度＝速い　and　加速度＝大きい　then　アクセル＝戻せ

複数個のファジィプロダクション規則から成り，規則の前件部の命題が連言による複合命題である場合のファジィ推論の結果は前節と同様な考え方から求めることが出来る。

i 番目のファジィプロダクション規則，すなわちファジィ条件命題は，ファジィ関係 R_i を用いて

$$「(x,\ y,\ z)\ \text{is}\ \ R_i」$$

と表すことができる。ファジィ関係 R_i は変数 x，y，z の空間を全体集合 X，Y，Z とすると，直積 $X\times Y\times Z$ 上のファジィ集合と考えられる。R_i として Mamdani の方法を用いると

$$R_i=(A_i\times B_i)\rightarrow C_i$$
$$=A_i\times B_i\times C_i$$

となる。

いま，入力をファジィ集合 A'，B' とすると，出力のファジィ集合 C'_i はファジィ関係の合成規則より

$$C'_i=(A'\times B')\circ R_i$$

として得られる。これをメンバーシップ関数で表すと

$$m_{c'}(z)=\bigvee_x\bigvee_y[(m_{A'}(x)\wedge m_{B'}(y))\wedge(m_{Ai}(x)\wedge m_{Bi}(y)\wedge m_{Ci}(z))]$$

となる。ここで，$m_{c'i}(z)$，$m_{A'}(x)$，$m_{B'}(y)$，$m_{Ai}(x)$，$m_{Bi}(y)$，$m_{Ci}(z)$ は，ファジィ集合 C'_i，A'，B'，A_i，B_i，C_i のメンバーシップ関数である。

ところで，制御においてはシステムの状態を定める変数 x，y は確定した数値（クリスプ値）として観測される。そこで，その値を x_o，y_o とすると，$x=x_o$，$y=y_o$ の時，$m_{A'}(x)=1$，$m_{B'}(y)=1$，また $x\neq x_o$，$y\neq y_o$ の時，$m_{A'}(x)=0$，$m_{B'}(y)=0$ と考えてよい。これを上記のメンバーシップ関数 $m_{c'i}(z)$ を求める式に代入すると，その式は次のように簡単になる。

$$m_{c'i}(z)=a_i\wedge m_{ci}(z)$$

ここで，a_i はファジィプロダクション規則 i の前件部に対する入力値 x_o，y_o の適合度であり，次の式で表すことが出来る。

$$a_i=m_{Ai}(x_o)\wedge m_{Bi}(y_o)$$

ところで各規則に関するファジィ条件命題は，それぞれ *OR* 結合されていると考えることができるので，各規則による総合的な推論結果はファジィ集合で表わすと

$$C'=\bigcup_{i=1}^{n}C'_i$$

となる。これをメンバーシップ関数で示すと

$$m_{c'}(z)=\bigvee_{i=1}^{n}m_{c'i}(z)$$

である。

以上のことから分かるように，全体としての推論結果は個々の規則による並列的な推論の *OR* 結合となっている。以上の推論の過程を図2に示す。

ところで，推論結果がファジィ集合で与えられるので，制御においてシステムを操作するには，ファジィ集合を確定値すなわちクリスプ値に変換する必要がある。これを非ファジィ化と呼ぶ。非ファジィ化の方法はいくつも提案されているが，ここでは次式によりファジィ集合 C' をそのメンバーシップ関数の重心座標 z_o の数値に変換する方法を採用する。

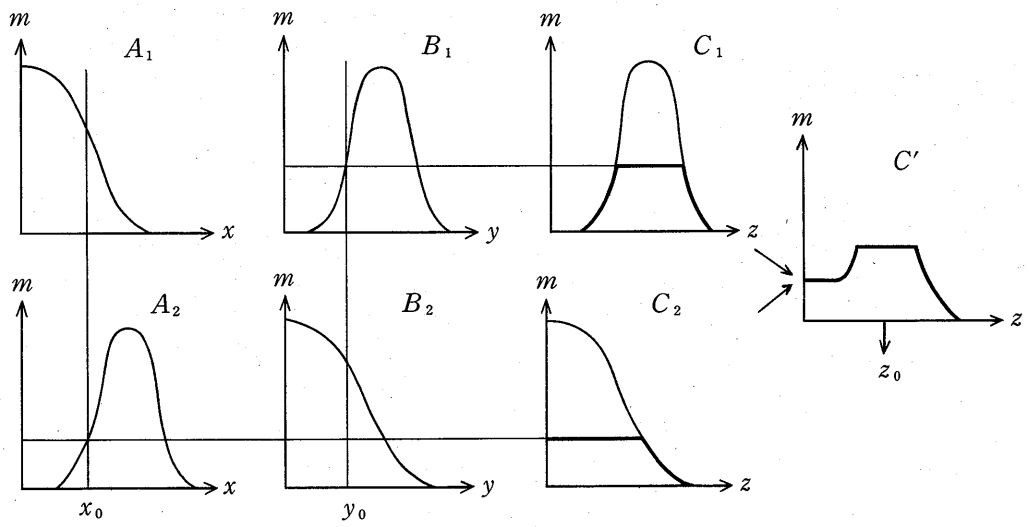


図2　ファジィ推論の過程

$$z_o = \frac{\int m_{c'}(z) z dz}{\int m_{c'}(z) dz}$$

このクリスプ値 z_o が，システムを操作する値となる。

2.4 ファジィ制御の手続き……まとめ

これまで述べたファジィ制御の手続きを以下にまとめておく。

(1) クリスプな入力値 x_o, y_o に対するファジィプロダクション規則 i の前件部の適合度 a_i を求める。

$$a_i = m_{Ai}(x_o) \wedge m_{Bi}(y_o)$$

(2) 適合度 a_i を規則 i の後件部に作用させて，規則 i の推論結果 $m_{ci}(z)$ を求める。

$$m_{c'i}(z) = a_i \wedge m_{ci}(z)$$

(3) 規則1から規則 n までがOR結合されているとして，n 個の規則により合成された推論結果 $m_{c'}(z)$ を求める。

$$m_{c'}(z) = \bigvee_{i=1}^{n} m_{c'i}(z)$$

(4) 推論結果として得られたファジィ値 $m_{c'}(z)$ を操作量としてのクリプス値 z_o に変換する。

$$z_o = \frac{\int m_{c'}(z) z dz}{\int m_{c'}(z) dz}$$

☆　　☆　　☆　　☆

なお，上記の(1), (2)の手続きにおけるmin演算（∧）では，適合度や重心の値がなめらかに変化しないこともあることが指摘されており，それに代わり通常の乗算を用いることも行われる。そこで，本シミュレーションにおいては，乗算を用いることにする。

3 ミサイルの追尾制御

3.1 シミュレーションの内容

ファジィ理論の有効性を認識するために，簡単なファジィプロダクション規則を定め，その規則を特徴付けるメンバーシップ関数を設定することにより，ミサイルが敵機を追尾し，それを撃墜することが出来るシミュレーションの例を取りあげる。ミサイルの防空空間は簡単な2次元空間とする。なお，このシミュレーションの研究は，ファジィ推論エンジンの特性を調べることが目的であり，特別な意図はない。

3.2 追尾のための規則

ミサイルが敵機を追尾する方法を，鬼ごっこ遊びで，鬼が子供達を追っかけて捕まえるやり方をもとに考えてみる（図3参照)。鬼ごっこでは，追われる子供は全く自由な方向に逃げることが出来るので，鬼は子供の逃げる方向だけを見定めて，自分の追っかけている方向の右手側に子供が見えれば，右手方向に向きを変えて追っかける。左手側に見えれば左手方向に向きを変えて追っかける。また前方に見えれば，そのままま っすぐに追っかける。そして鬼が追われる子供よりも速く追っかけることが出来れば，これだけの動きで必ず子供を捕まえることが出来る。我々はこのことを経験から学んでいる。ミサイルが敵機を追尾し撃墜する方法にも，このような簡単な経験則を適用することができるはずである。

そこでこの経験則をミサイルが敵機を追尾するためのファジィプロダクション規則に翻訳する

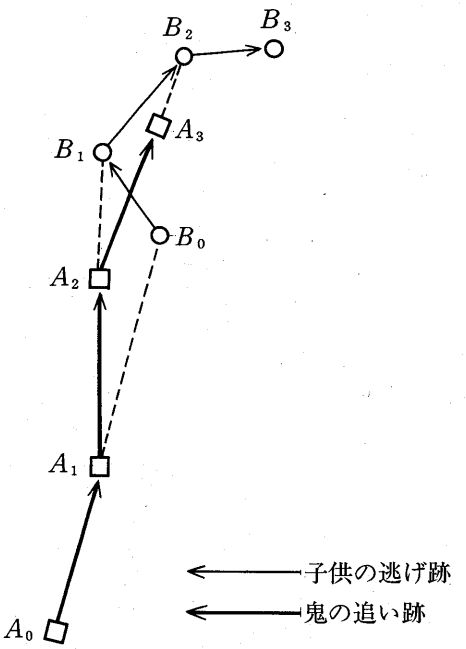


図3　鬼ごっこにおける追い方

と，以下のような簡単な三つの規則になる。

規則1：　if　敵機が前方「右手方向」に存在する
　　then　ミサイルは「右手方向」に飛行角度をとれ
規則2：　if　敵機が前方「ほぼそのままの方向」に存在する
　　then　ミサイルは「ほぼそのままの方向」で飛行せよ
規則3：　if　敵機が前方「左手方向」に存在する
　　then　ミサイルは「左手方向」に飛行角度をとれ

3.3　追尾アルゴリズム

ミサイルが敵機を追尾するアルゴリズムは，次の通りである（図4参照）。ある時刻 t_0 に敵機の存在する位置 e_0 に機首を向けて飛行するミサイルは，次の時刻 t_1 には，敵機が位置 e_1 に移動するので，その位置の方向に向けて機首を θ_0 度回転して，飛行しなければならない。ミサイルが位置 m_1 に達したら，敵機は位置 e_1 から e_2 に移動するので，再度ミサイルの機首を位置 e_2 にむけて……という手続きを繰り返すことにより，ミサイルは敵機に接近することができる。このときのミサイルの機首の追尾角度 θ の累計 Θ は

$$\Theta = \theta_0 + \theta_1 + \theta_2 + \theta_3 + \cdots\cdots$$

となる。そして，それぞれの追尾角度 θ_i が正確であり，ミサイルが敵機より速ければ，最終的にミサイルは敵機を撃墜することができる。

現実の場合，測定される追尾角度は誤差を持っている。上記の追尾アルゴリズムを誤差を伴っても正しく追尾出来るように改良しよう。図5に示すように，敵機が位置 e_0 の時，m_0 に位置するミサイルが正しく機首を位置 e_0 に向けているとする。そして，敵機が位置 e_1 に移動した時の

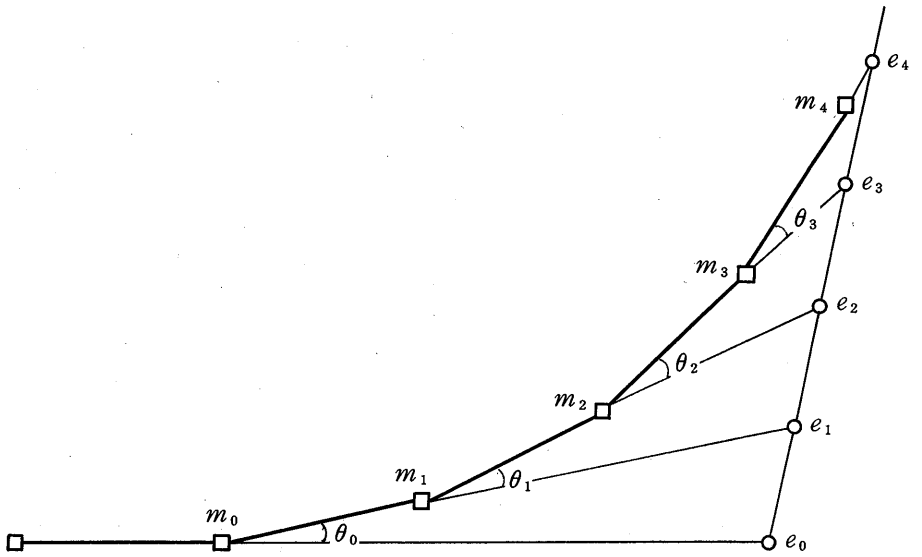


図 4　敵機の追尾方法

追尾角度に誤差が生じ，ミサイルの位置が $m_{1'}$ ではなく，m_1 になったとする。この $m_{1'}$ の位置から，次の時刻での敵機の位置 e_2 に機首を向ける追尾角度 θ は θ_1 ではなく，実際には m_0 と $m_{1'}$ との延長線と $m_{1'}$ と e_2 の線分のなす角 $\theta_{1'}$ となる。従って，m_0 の位置から m_2 の位置にミサイルが飛行する累計の追尾角度 Θ は誤差がないときは

$$\Theta = \theta_0 + \theta_1$$

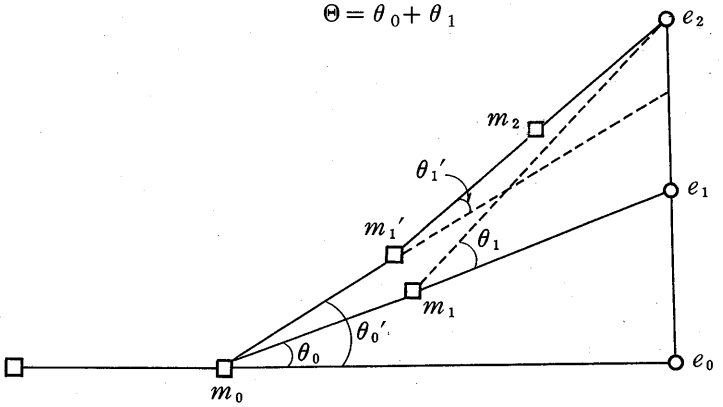


図 5　誤差補正の方法

であるが，誤差があるときは

$$\Theta = \theta_0 + \theta_{1'}$$

となる。このとき，$\theta_0 \geqq \theta_{0'}$ の時 $\theta_1 \leqq \theta_{1'}$ であり，$\theta_0 \leqq \theta_{0'}$ の時 $\theta_1 \geqq \theta_{1'}$ となる。すなわち累計の追尾角度 Θ は位置 $m_{1'}$ で生じた誤差を吸収している。このような誤差の補正を追尾の手続き毎に行うのである。

3.4 ファジィ推論エンジン

ミサイルの飛行制御におけるファジィ推論エンジンの役目は，以下に示すように視角距離 l をクリスプな入力値として，これをもとにファジィプロダクション規則を用いたファジィ推論により，追尾角度 θ を求めることである（図6）。

$$\theta \leftarrow l$$

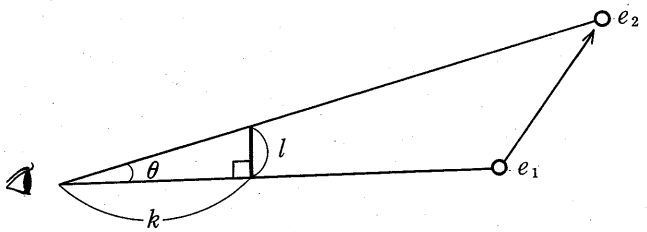
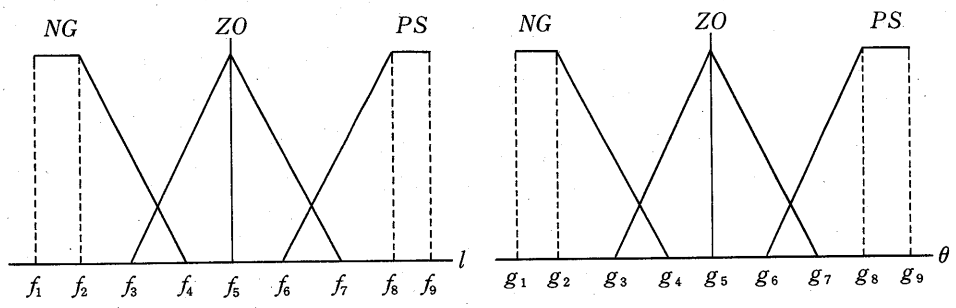


図6　視角距離の意味

視角距離 l とは，図6に示すようにミサイルの飛行方向から敵機がどのくらい離れて（ずれて）いるかの値である。この値が言語ラベル「右手方向」，「左手方向」，「ほぼそのままの方向」の付いたファジィ集合を定義するメンバーシップ関数の独立変数の値に対応する。具体的には視角距離 l は，図中の長さが一定値である線分 k を基準にして，その長さ（距離）が定められる。実際には k の値は1が用いられる。ファジィプロダクション規則の前件部と後件部で使用されるファジィ変数（言語ラベル）は，図7に示すような三角型と部分台形型のメンバーシップ関数で表わされるものとする。言語ラベルは，PS（positive），Z0（zero），NG（negative）の略号を用いている。



(a) 前件部　　(b) 後件部

図7　追尾制御のためのメンバーシップ関数

3.5 シミュレーション条件の設定

ミサイルは敵機との視角距離を測定する装置，視角距離から追尾角度を推論するファジィ推論エンジン，および追尾角度をもとに自動的に敵機を追尾する装置を搭載しているものとする。敵機は任意の地点から一定速度のもとで侵入し，ミサイルは敵機発見後，指定された時間経過の後，指定された速度で自動的に発射される。ミサイルは敵機との視角距離を測定しながら敵機を自動追尾し，両飛行物体が衝突することにより撃墜成功となる。

ミサイルと敵機の形状は円盤状とする。パソコンによるシミュレーションなので処理速度の関係からミサイルと敵機を同時に動かすことが出来ないので，これらの飛行物体を交互に動かすこととする。ここではミサイルと敵機が交互に1回づつ移動する1サイクルの時間を1単位時間（1 ut）と呼ぶ。シミュレーション画面の例を図8に示す。図中に記載された数値は防空空域の座標であり，その単位の距離を1単位距離（1 ud）と呼ぶ。そして1 ut当り1 ud移動するのを1単位速度（1 uv）と呼ぶ。

シミュレーションのための画面からの入力データは次の通りである。

```
＞type, speed 1, speed 2, time ？
＞x, y, sign ？
```

type　：1……ファジィ制御（飛行物体は円盤状で軌跡なし）
　　a……ファジィ制御（飛行物体は点で軌跡あり）
　　2……直接制御（飛行物体は円盤状で軌跡なし）
　　b……直接制御（飛行物体は点で軌跡あり）
speed 1：ミサイルの速度
speed 2：敵機の速度
time　：敵機発見後のミサイル発射までの時間

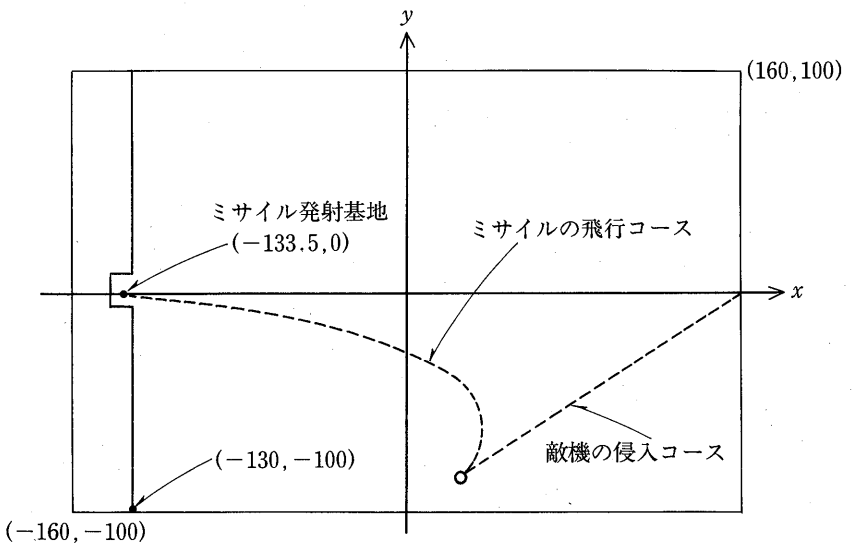


図8　追尾制御のシミュレーション画面

x, y　：敵機の侵入コース
　　x 軸と y 軸の座標を与える（図 8 参照）
sign　：敵機の x 軸に対する飛行方向
　　正方向……＋1
　　負方向……－1

直接制御とは追尾角度にピタゴラスの定理から幾何学的に求められる理論角度を適用するものであり，ファジィ制御と比較・参考にするために設けるものである。なお，ミサイルと敵機の大きさ（サイズ）はプログラム内部のパラメータとして与えるものとする。

3.6 結　　果

プログラムは Advanced RUN/C（Version 3.02）を用いて作成し，PC-9801（VM）のもとで実行した。プログラムを走らせ，ファジィ推論から導かれたミサイルの追尾角度，すなわちファジィ角度 θ がピタゴラスの定理から求められる理論角度 α について，次式

$$\theta \fallingdotseq \alpha \qquad (-50^\circ \leqq \alpha \leqq 50^\circ)$$

がだいたい成立するように，メンバーシップ関数の形状を支配するパラメータの値を定めた。“だいたい”でよいというのは，3.2 節で述べたように本シミュレーションにおける追尾アルゴリズムが，1 サイクルの追尾ごとに生ずる追尾角度の誤差を吸収することが出来るからである。なお，パラメータの値は視角距離 l に関して，$|l| \leqq 0.3$，$0.3 < |l| \leqq 0.6$，$|l| > 0.6$ でそれぞれ異なった値を採用した。パラメータの値を表 1 に示す。表中の小中大の区分は，上記の視角距離 l の絶対値の大きさの区分に対応している。また，表中では，メンバーシップ関数の形状は m 軸（通常の y 軸）に関して対称なので，独立変数が正またはゼロの場合のパラメータしか示していない。なお，ミサイルと敵機の大きさは 0.5ud の値を採用した。

ファジィ制御における追尾シミュレーションの実行結果をいくつか表 2 に示す。実行例 1 の意味を説明しよう。ミサイルと敵機はそれぞれ 20uv，10uv の速度で飛行，敵機発見後 2 ut 後にミサイルを発射する。敵機は x 軸が －50ud の位置から y 軸が 70ud の位置へ向かって（飛行方向は＋1）侵入する。この条件のもとで，結果は，ミサイルの飛行距離は 193.09ud であり，その時の追尾の総理論角度に対する総ファジィ角度の平均誤差（推論角度誤差率）が0.80%，理論撃

表 1　追尾制御のためのメンバーシップ関数のパラメータ

		f_5	f_6	f_7	f_8	f_9
視角距離	小	0	0	0.9	1.0	999
	中	0	0	1.0	1.13	999
	大	0	0	1.1	1.1	999
		g_5	g_6	g_7	g_8	g_9
追尾角度	小	0	0	40	60	78
	中	0	0	40	60	79
	大	0	0	40	60	76

表2　追尾シミュレーションの結果

	ミサイル速度	敵機速度	発射時間	敵機侵入コース			撃墜距離	飛行距離	撃墜誤差率	推論角度誤差率
				x軸	y軸	方向				
実行例1	20	10	2	−50	70	+1	0.01	193.09	0.01	0.80
実行例2	20	12	0	88	90	−1	0.25	296.19	0.08	2.50
実行例3	18	12	2	−120	10	+1	0.06	175.60	0.03	12.29
実行例4	12	8	3	0	−10000	−1	0.05	177.47	0.03	0.29
実行例5	10	5	0	−1000	40	−1	0.02	195.24	0.01	5.04

墜位置から外れた距離（撃墜距離）は0.01ud，飛行距離に対する撃墜誤差率が0.01％であるということを示している。

実行例3では，ファジィ推論エンジンによる角度推定能力が限界を越えるため，正確に求められていない。それでも撃墜誤差率が0.03％で撃墜に成功している。これは追尾アルゴリズムが誤差を吸収することを示している。他の多くの実行例についても，この表に示した結果と大体同様である。特に撃墜誤差率は1000分の1から10000分の1のオーダーであり，非常な高精度で撃墜することが出来たといえる。

4　自動車の走行制御

4.1　シミュレーションの内容

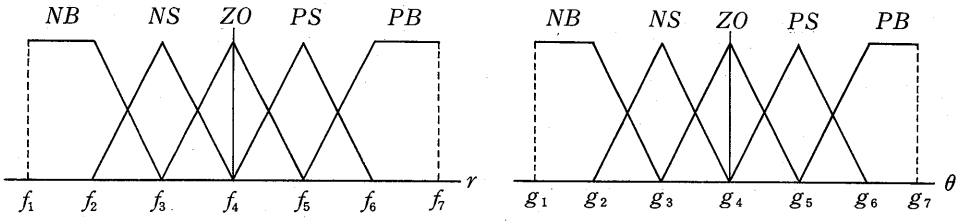
自動車をテストコースの中央位置上で速度を一定に保って走行させ，与えられた停止線上で自動的に止めるシミュレーションである。すなわち，ファジィ推論による位置制御，速度制御，停止制御を行う。これらの制御は相互に関連し合っている。

4.2　位置制御

位置を制御するモデルは，「テストコースの中央にガイドウエイ（目標位置）が設定されており，自動車はガイドウエイからの距離を測定し，その距離からのズレに応じて進行方向の角度を決定する」というものである。このため経験則としてのファジィプロダクション規則は，次の5つを用いることとする。

規則1：　if　目標位置が「右手横方向遠く」に存在する
　　then　「右手方向に大きく」回転して進め
規則2：　if　目標位置が「右手横方向近く」に存在する
　　then　「右手方向に小さく」回転して進め
規則3：　if　目標位置が「ほぼそば」に存在する
　　then　「ほぼそのまま」進め
規則4：　if　目標位置が「左手横方向近く」に存在する
　　then　「左手方向に小さく」回転して進め
規則5：　if　目標位置が「左手横方向遠く」に存在する
　　then　「左手方向に大きく」回転して進め

現時点iの車の位置（x_i, y_i）と目標位置（x_o, y_o）からズレの距離r_iを求め，そのズレから上



(a) 前件部　　(b) 後件部

図9　位置制御のためのメンバーシップ関数

記の規則を用いて車の進行角度 θ_i をファジィ推論して求め，その時の v_i を用いて，次の時点 $i+1$ の車の位置（x_{i+1}, y_{i+1}）を算定する。

$$r_i=\sqrt{(x_i-x_o)^2+(y_i-y_o)^2}$$
$$\theta_i \leftarrow r_i$$
$$x_{i+1}=x_i+v_i \cdot \cos\theta_i$$
$$y_{i+1}=y_i+v_i \cdot \sin\theta_i$$

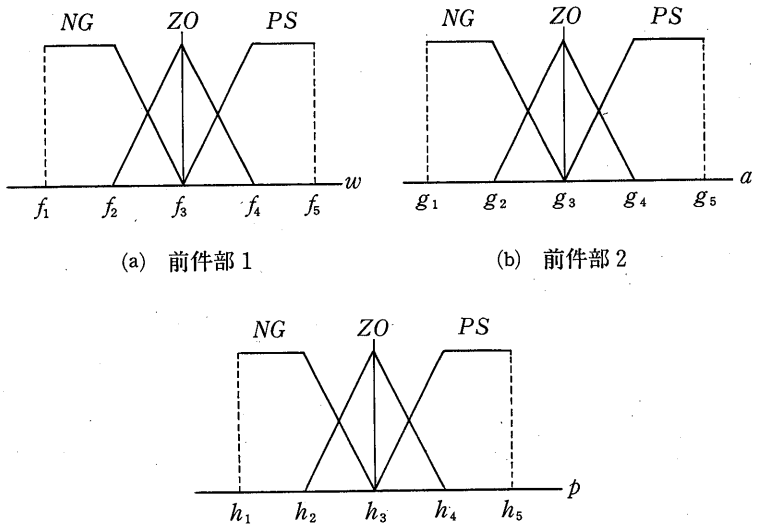
位置制御で用いるファジィプロダクション規則の前件部と後件部で使用されるファジィ変数（言語ラベル）は，図9に示すような三角型と部分台形型のメンバーシップ関数で表わされるものとする。言語ラベルは，PB（positive big），PS（positive small）と，Z0（zero），NS（negative small），NB（negative big）の略号を用いている。

4.3 速度制御

速度の制御は加速機（アクセル）の働きをモデル化する。すなわち，目標速度と加速度に応じ，加速機を踏んだり緩めたりして速度を調節するものとする。このため経験則としてのファジィプロダクション規則は次の7つを用いるが，これは山川の「倒立振子」の経験則を速度調節に合うように簡易化したものである。

規則1：　if　目標速度を越えており，加減速はほぼない
　　then　加速機を緩めよ
規則2：　if　目標速度を越えており，加速されている
　　then　加速機を緩めよ
規則3：　if　目標速度を越えており，減速されている
　　then　加速機はほぼそのままにせよ
規則4：　if　目標速度にほぼ達しており，加減速はほぼない
　　then　加速機はほぼそのままにせよ
規則5：　if　目標速度に達しておらず，加速されている
　　then　加速機はほぼそのままにせよ
規則6：　if　目標速度に達しておらず，減速されている
　　then　加速機を踏み込め
規則7：　if　目標速度に達しておらず，加減速はほぼない
　　then　加速機を踏み込め

現時点 i における速度 v_i と目標速度 v_o との差 w_i，およびその時点での加速度 a_i を求め，それ



(a) 前件部 1　　(b) 前件部 2

(c) 後 件 部

図10 速度制御のためのメンバーシップ関数

らの値に上記の規則を適用して次の時点 $i+1$ のための加速機の出力 p_{i+1} をファジィ推論し，その出力から加速速度（$k \times p_{i+1}$ …… k は加速定数）を求め，その時点の速度 v_{i+1} を算定する。

$$w_i = v_i - v_o$$

$$a_i = v_i - v_{i-1}$$

$$p_{i+1} \leftarrow w_i \quad \text{and} \quad a_i$$

$$v_{i+1} = v_i + k \cdot p_{i+1}$$

速度制御で用いるファジィプロダクション規則の前件部と後件部で使用されるファジィ変数（言語ラベル）は，図10に示すような三角型と部分台形型のメンバーシップ関数で表わされるものとする。言語ラベルは，PS（positive），Z0（zero），NG（negative）の略号を用いている。

4.4 停止制御

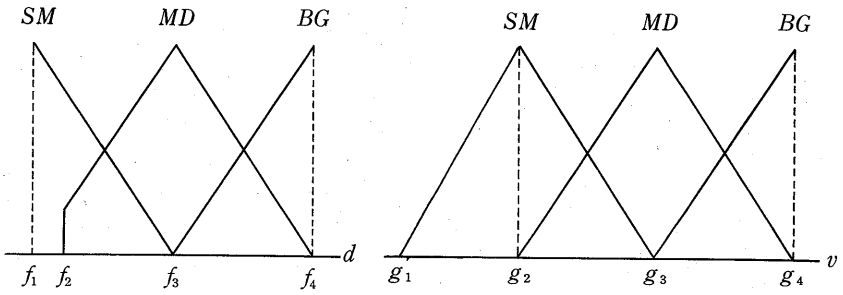
停止を制御するモデルは「停止区間内に車が進入するとブレーキが働き，停止線から距離に応じて徐々に減速し，停止線直前またはその上で完全に停止する」というものである。このため経験則としてのファジィプロダクション規則は，次の4つを用いることとする。

規則 1 ： if　車は停止区間の「開始位置付近」に存在する
　　then　速度を「やや」落とせ

規則 2 ： if　車は停止区間の「真中付近」に存在する
　　then　速度を「だいぶ」落とせ

規則 3 ： if　車は停止線に「十分近い所」に存在する
　　then　速度を「ほとんど」落とせ

規則 4 ： if　車は停止線の「直前の所」に存在する
　　then　速度を「ゼロ」にせよ



(a) 前件部　　(b) 後件部

図11　停止制御のためのメンバーシップ関数

規則4の場合，後件部の命題が「速度がゼロ」というクリスプ命題となっている。この規則は，実際のメンバーシップ関数の設定時においては規則3で代用することにする。従って，規則3の後件部の「速度をほとんど落とせ」というファジィ命題のファジィ集合に対するメンバーシップ関数の形状には工夫を凝らす必要がある。これについては後で詳しく述べる。

現時点 i の車の位置（x_i, y_i）と停止線の位置（x_o, y_o）を用いて車の停止線からの距離 d_i を求め，その距離に上記の規則を適用し，次の時点 $i+1$ の車の速度 v_{i+1} をファジィ推論する。

$$d_i = x_o - x_i$$

$$v_{i+1} \leftarrow d_i$$

停止制御で用いるファジィプロダクション規則の前件部と後件部で使用されるファジィ変数（言語ラベル）は，図11に示すような三角型と部分三角型のメンバーシップ関数で表わされるものとする。言語ラベルは，SM（small）と，MD（medium），BG（big）の略号を用いている。

ファジィ制御のもとで，自動車を完全に停止させるには，上述したようにメンバーシップ関数の形状に工夫を凝らす必要がある。というのは，完全な三角型のメンバーシップ関数を用いた場合には，完全に停止（速度ゼロ）するまでに時間がかかりすぎるか，またはパラメータの設定を多少変えても停止線のかなり手前で停止してしまうかである。ファジィ推論エンジンが停止線直前またはその上で妥当な時間内で速度ゼロを推論するには，図11に示すように前件部の言語ラベルMDの三角型のメンバーシップ関数の一部分を（$0 \leqq x \leqq s$）の間で

$$m(x) = 0$$

とするのである。s の値は停止区間距離の1000分の15とする。ただし後件部の言語ラベルSMのメンバーシップ関数は三角型とし，その積分結果がゼロとなるようにする。

このような工夫により，停止線手前 s の距離付近で，確実に速度はゼロとなり，停止する。しかし，これでは s の値如何では，停止線直前といえない停止が頻発するであろう。そこで，ファジィ推論エンジンに与える停止区間距離の値をシミュレーション開始時に外部から与える停止区間距離に $c \times s$ の値を加えた値とすることにする。すなわち，停止線をファジィ推論エンジンのもとではその値の分だけを実質的に先に延ばすことにする。この c は1以下の数であり，ここではこれを停止係数と呼ぶ。停止係数 c によっては甚だわずかではあるが停止線を越えることも生じるかもしれないが，これはシミュレーション結果を見定めて評価することとする。

なお，実際の自動車の場合，停止区間が短い場合には，急ブレーキをかけても止まらない。本シミュレーションにおいても，停止区間内において進入速度が速すぎる場合には，車は停止線を

オーバーすることになる。そこで，このような場合が生じたら，停止線を形式的に進行方向側に延ばして設定し直し，停止制御を再度実行して停めることにする。

4.5 シミュレーション条件の設定

自動車は，①位置制御するために目標位置からのズレの距離を測定する装置，およびズレの距離をもとに進行方向を推論するファジィ推論エンジン，②速度制御するために目標速度からのズレの速度と加速度をもとに加速機の出力度合を推論するファジィ推論エンジン，および出力度合により速度を加減速する加速機，③停止制御のために停止位置からの距離を測定する装置，および停止位置からの距離をもとに速度を推論するファジィ推論エンジン，等の装置を搭載しているものとする。これらの装置が連動して自動車の走行を制御する。

自動車の形状はパソコンによるシミュレーションなので処理速度を考慮して点とし，その走行軌跡を画面に表示する。シミュレーションの1サイクルを駆動する時間を1単位時間（1 ut）と呼ぶことにする。シミュレーション画面の例を図12に示す。図中に記載された数値はテストコースの座標と距離であり，その単位を1単位距離（1 ud）と呼ぶ。そして1 ut当り1 ud移動するのを1単位速度（1 uv）と呼ぶ。

シミュレーションのための画面からの入力データは次の通りである。

```
＞type, speed 1, speed 2, interval ？
＞x, y ？
```

type　：1……単純ファジィ制御
　　2……複合ファジィ制御
　　3……クリスプ制御
speed 1　：目標速度1
speed 2　：目標速度2
interval　：停止区間距離
x, y　：自動車のスタート位置
　　座標を与える（図12参照）

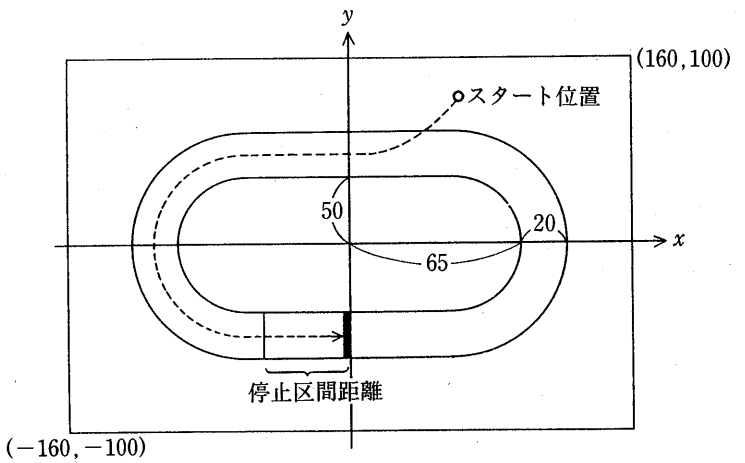


図12　走行制御のシミュレーション画面

前節までに述べたファジィ制御が上記の type の項の複合ファジィ制御である。単純ファジィ制御はメンバーシップ関数の個数の少ない初歩的な制御であり，クリスプ制御はプロダクション規則にクリスププロダクション規制を適用した制御であるが，これらは複合ファジィ制御と比較するために別途作成して設定したものである。

なお，加速定数 k，停止係数 c の値は，プログラム内部のパラメータとして与えるものとする。

4.6 結　　果

プログラムは Advenced RUN/C（Version 3.02）を用いて作成し，PC-9801（VM）のもとで実行した。プログラムを走らせ，位置制御，速度制御，停止制御が充分うまく働くように，メン

表 3　位置制御のためのメンバーシップ関数のパラメータ

距離	f_4	f_5	f_6	f_7
	0	20	40	50
角度	g_4	g_5	g_6	g_7
	0	36	72	90

表 4　速度制御のためのメンバーシップ関数のパラメータ

速度差	f_3	f_4	f_5
	0	3	999
加速度	g_3	g_4	g_5
	0	2	999
加速機	h_1	h_2	h_3
	0	1.5	3.5

表 5　停止制御のためのメンバーシップ関数のパラメータ

位置	f_1	f_2	f_3	f_4
	0	$0.015d$	$0.5d$	d
速度	g_1	g_2	g_3	g_4
	$-0.5v$	0	$0.5v$	v

表6　走行シミュレーションの結果

	目標速度1	目標速度2	停止間隔	スタート位置		停止位置	走行距離	走行時間
				x	y			
実行例1	8.50	6.20	15.2	80	80	0.006	381.90	61
実行例2	11.11	13.33	18.0	0	70	−0.045	302.52	33
実行例3	8.55	10.88	15.3	0	50	0.023	296.35	40
実行例4	9.30	11.20	14.5	0	0	−0.001	316.94	40
実行例5	10.30	6.70	20.5	−150	10	0.071	175.53	34

バーシップ関数の形状を決定するパラメータの値を定めた。そのパラメータを表3，表4，表5に示す。表3，表4については，メンバーシップ関数の形状はm軸（通常のy軸）に関して対称なので，独立変数が正又はゼロの場合のパラメータしか示していない。表5中のdは停止区間距離，vは停止区間進入時の速度を意味する。なお，加速機の出力pは最大値が1，最小値が0になるように最終的に規格化した。また，加速定数kは2 uv/utの値，停止係数cは0.666の値を採用した。

複合ファジィ制御における走行シミュレーションの実行結果をいくつか表6に示す。実行例1の意味を説明しよう。自動車はまず目標速度8.5uvに向けて走行し，それが達成されたら次に目標速度6.2uvに向けて走行する。その速度を交互に繰り返し，停止線の手前15.2ud以内に達したら，停止制御が稼働する。自動車は座標（80ud，80ud）の位置からスタートする。この条件のもとで，結果は走行距離は381.90ud，走行時間は61ut，停止線手前0.006udの位置で停止したということを示している。他の多くの実行例についても，この表に示した結果と大体同様である。

一般に位置制御に関しては，画面で判断する限り，テストコースの中央位置を保たせながら車の走行をスムーズにコントロールすることが出来た。速度制御に関しては，目標速度を全く誤差なく正確に達成することが出来た。また，停止制御に関しては，停止区間進入時点における速度の1 ut当りに進む距離の1から1.5倍の停止区間距離があれば，ほとんどの場合，徐々に減速しながらほとんど停止線上といえる位置で停止することが出来た。これらの結果は非常な高精度でファジィ制御による走行制御が働いていることを示している。

本ファジィ制御と同様な駆動機構のもとで，クリスププロダクション規則（位置，速度，停止の各制御における規則数は対称性も考慮すると，それぞれ23個，21個，11個）を用いたクリスプ走行制御を行ってみたが，ジグザグ走行となり，目標速度の正確な達成は不可能であり，しかも停止線直前での停止は困難であった。

5　おわりに

本論文で示したミサイルの追尾と自動車の走行をシミュレートするファジィエキスパートシステムは，日本ファジィ学会主催「第6回ファジィシステム・シンポジュウム（1990）」のデモン

ストレーション部門で公開したものである。本システムは簡単なファジィ理論の応用ではあるが，この理論の初心者である筆者も，このシステムの作成を通じてファジィ理論の有効性を見定めることが出来たといえる。今後はファジィエキスパートシステムの考え方を信用評価，経営分析等の経営管理分野に応用しようと思っている。なお，本研究の一部は1990年度文教大学共同研究費によって行われたものであり，御援助くださった関係の方々に深くお礼申し上げたい。

参 考 文 献

(1) 菅野道夫著「ファジィ制御」日刊工業新聞社，1988年
(2) 寺野寿郎編「応用ファジィシステム入門」オーム社，1989年（第2章　応用のためのファジィ集合論（広田薫著））
(3) 本多中二・大里有生著「ファジィ工学入門」海文堂，1989年
(4) 向殿政男著「ファジィのはなし」日刊工業新聞社，1989年
(5) 山川　烈著「Fuzzy コンピュータの発想」講談社，1988年
(6) 広内哲夫・宮川裕之著「シミュレーションゲームへのファジィ理論の適用」　第6回ファジィシステム・シンポジュウム講演論文集，日本ファジィ学会，1990年