SYNCHRONIZATION OF AUTONOMOUS OBJECTS IN DISCRETE EVENT SIMULATION

Ralph V. Rogers
Assistant Professor
Department of Industrial Engineering and Management Systems
University of Central Florida
P.O. 25000
Orlando, FL
32816-0450

ABSTRACT

Autonomous objects in event-driven discrete event simulation offer the potential to combine the freedom of unrestricted movement and positional accuracy through Euclidean space of time-driven models with the computational efficiency of event-driven simulation. The principle challenge to autonomous object implementation is the object synchronization. The concept of a spatial blackboard is offered as potential methodology for such synchronization. The issues facing implementation of a spatial blackboard are outlined and discussed.

INTRODUCTION

Autonomous objects in simulation models can alter their temporal and spatial goal trajectories conditional upon their current state and the state of the rest of the system. Autonomous objects have been generally associated with continuous or time-driven discrete simulation models. However, in highly dynamic, complex environments involving multiple autonomous objects such as air traffic control, the resource demands of such models generally limit their use. Use of autonomous objects in the more computationally efficient event-driven discrete event simulation faces several challenges. In particular, discrete-event simulation environments with multiple autonomous objects requires strategies to establish common reference frames (i.e. synchronization) for object decision making. This paper outlines research into the synchronization methods for autonomous objects in object-oriented, event-driven, discrete event simulation.

OBJECT SYNCHRONIZATION

In traditional discrete event simulation, the two established methods of simulation models are event-driven and time-driven model. In time-driven simulation modeling, time is advanced in fixed increments. With every advance of the clock, each entity and process updates its state to reflect the new time of the simulation clock. Conditional operational decisions are made (and synchronized) at these fixed time intervals. With the fixed time method, the state of each object in the simulation must be updated, the conflicts and resolution

(NASA-CR-189534) SYNCHRONIZATION OF AUTONOMOUS OBJECTS IN DISCRETE EVENT SIMULATION (University of Central Florida) 33 p CSCL 12B

N92-15666

Unclas
G3/66 0061518

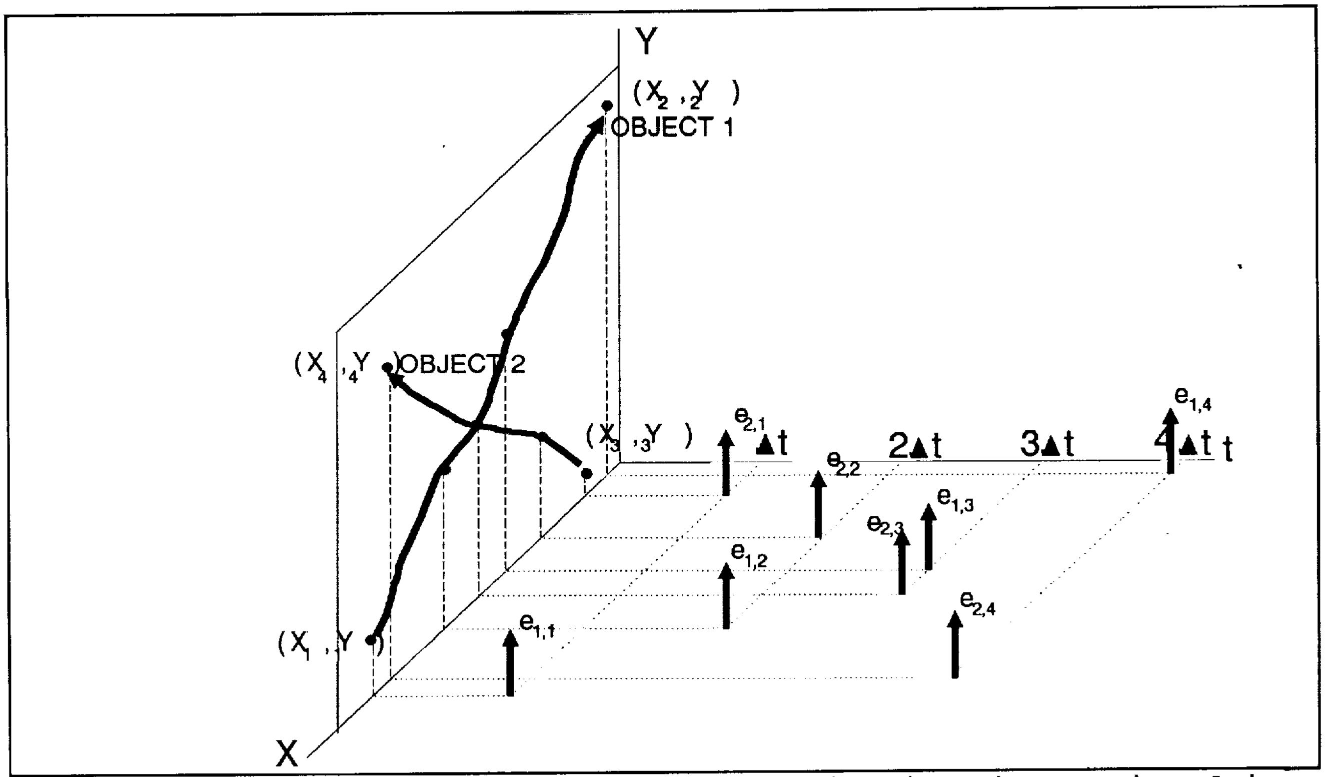


Figure 1. Object movement and synchronization in a time-driven simulation model.

identified, and actions implemented at each increment of time. The reference for synchronization of actions is the common time defined by the increment. Figure 1 provides a graphical illustration of this approach for two autonomous objects moving through a plane. Notice that each object must establish its relationship to the other object at each clock increment Δt.

In event-driven simulation modeling, the next scheduled event (i.e. state change) defines the next increment of time that advances the clock. Thus, objects may have their states updated at different times and the resulting increments between time advances may vary widely through the course of a simulation exercise. However, a common reference is still required to resolve possible conflicts and other interactions between objects (i.e. synchronization). Such synchronization of object interactions and dependencies in the modeled system requires operational decision points. These decision points are not generally fixed in time as in time-driven simulation. Instead, they are more commonly conceived as fixed in space. That is, the event-driven solution to synchronization is to fix the decision points in Euclidean space and thereby fixing the spatial increments of the model.

The classic paradigm for event-driven modeling which typifies this approach to synchronization is a network of nodes and arcs. Nodes are used to represent decision points and arcs to represent specific distances and/or time. Once an object starts to transverse an arc, it will complete the arc in some specified time (usually a random variable). This time may be associate with travel time for the specified distance or simply the service time. Conflicts are resolved at the nodes. Indeed, an object (or entity) typically will wait at a node until sufficient conditions exist for it to proceed onto the next arc. Thus, the reference point for event driven modeling is the fixed position nodes or similar simulation construct corresponding to fixed points in Euclidean space.

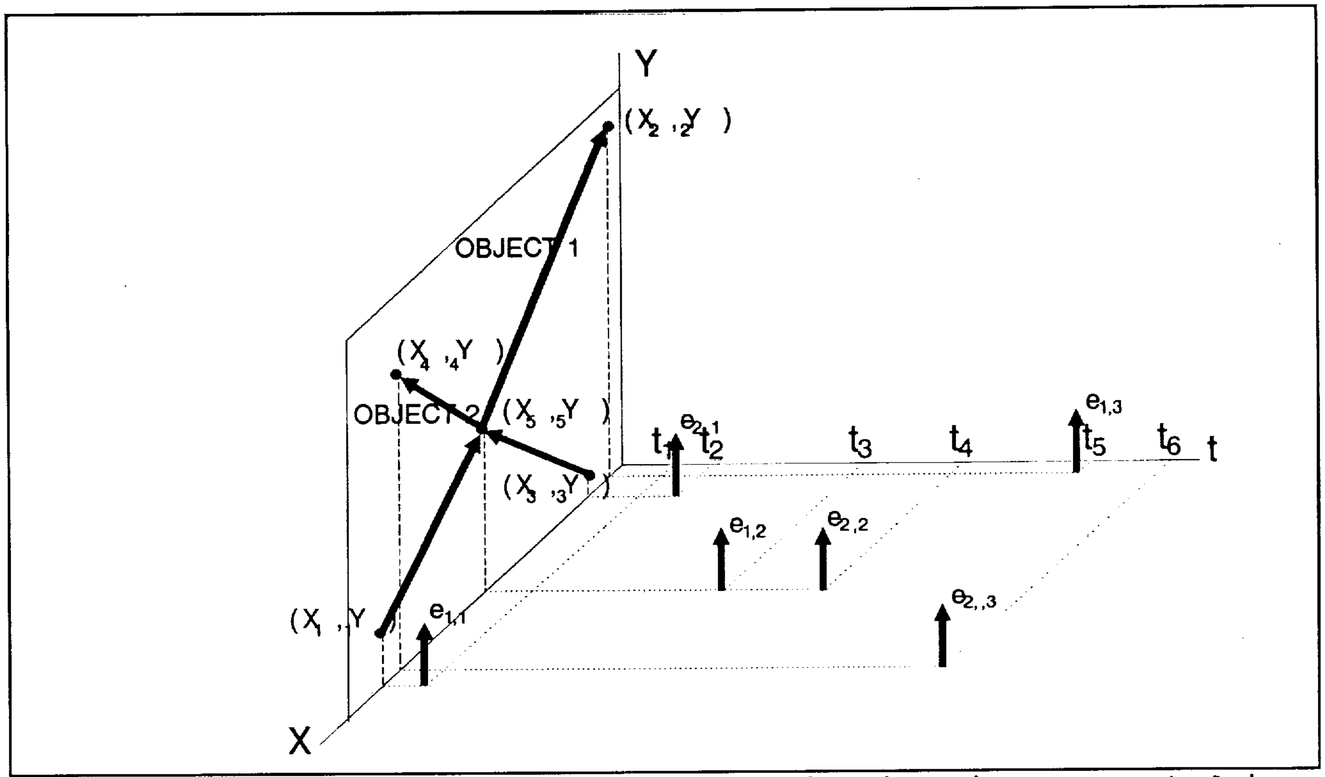


Figure 2. Object movement and synchronization in a event-driven simulation model.

Figure 2 provides a graphical illustration of the corresponding event-driven simulation of two objects moving on a plane. Note that the trajectory paths are explicitly defined as part of the model. The synchronization point for the two objects is point X_5,Y_5 of X-Y plane. In this case, the decision or control logic must only consider the objects and interactions associated with and prior to events $e_{1,2}$ and $e_{2,2}$.

Figure 2 also reveals the paradox confronting autonomous object use in event-driven simulation. The objects as autonomous must have the freedom to modify their trajectory through Euclidean space, to go where their goals and operational rules specify. However, to synchronize the actions of these or any objects, some common reference point must be established. Figure 3 illustrates this problem for autonomous objects in an event driven simulation. In this example the objects schedule their next event (i.e. an arrival) some time in the future (t_3 and t_4). However, with no common points defined for them in the model in either time or space, there is no mechanism for synchronization. What, if any, interactions must be accommodated between the two objects are simply not captured in the model. Each object proceeds unaware and unaffected by the other regardless of each's possible trajectory. Clearly, this not a viable basis on which to build a simulation modeling approach.

What is desired is simulation modeling based on autonomous objects which can combine the freedom of unrestricted movement and positional accuracy through the Euclidean space associated with time-driven models and the computational efficiency of event-driven models. To achieve such capabilities in event-driven simulation modeling, each autonomous object must be allowed to schedule its next event anywhere in the Euclidean environment of the model its operational rules permit. If no other impeding events happen between the time an object schedules the next event and the time that scheduled event occurs, the object's state change occurs as scheduled. However, if other events occur during this period that effects the behavior

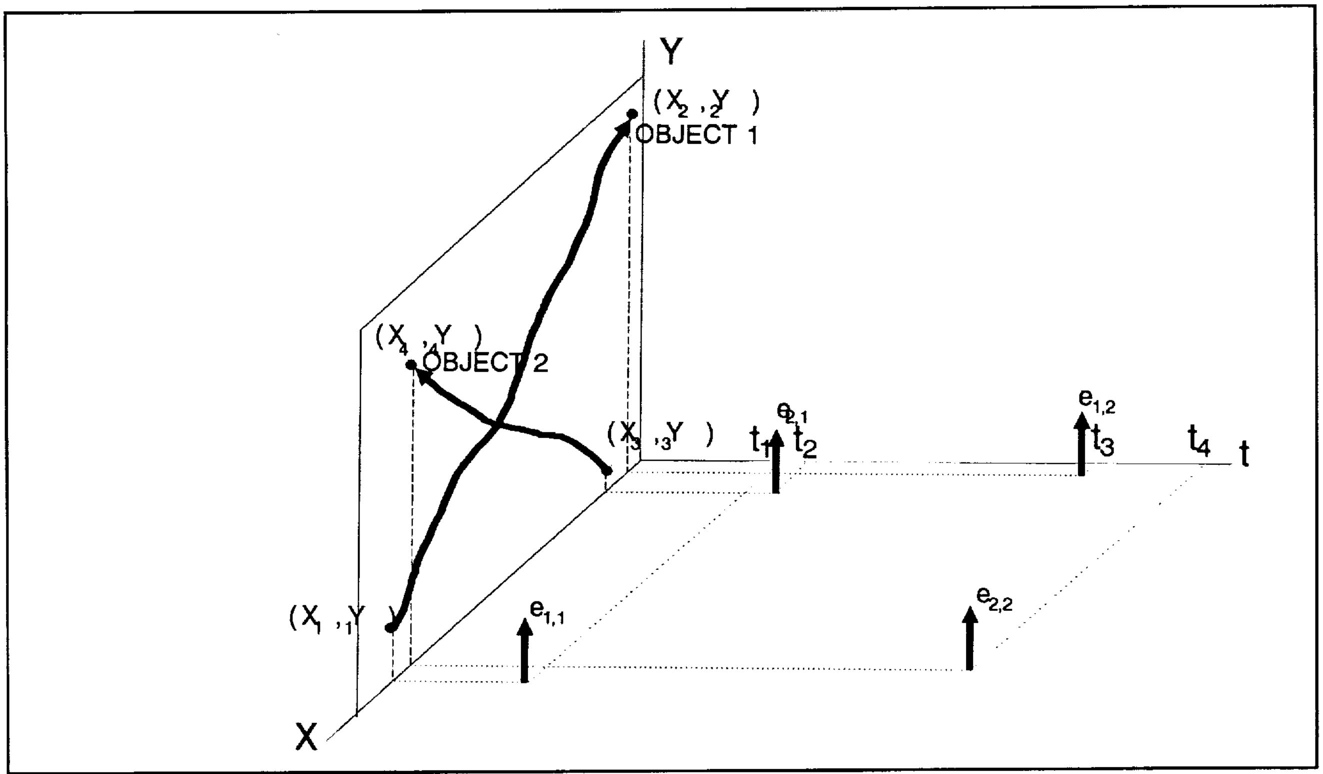


Figure 3. Autonomous Objects in event-driven discrete event simulation.

of the object and its goal trajectory, the object is notified of the potential conflict. The object may then take action to schedule a change of state (i.e. an event) to avoid the potential conflict or precipitate a scheduled event modification in one or more other objects. How objects recognizes and communicates a potential conflict and how they schedule their next state change are the major issues of investigation in the author's current research project. General strategies for these issues are discussed in the following.

CONFLICT STRATEGIES

The simplest strategy for addressing the event scheduling and conflict recognition problem would be for each object to schedule its next event as would best meet its own objectives. The object would then "ask" every other objects in the object model if this event and resulting actions would cause a spatial or potential spatial conflict with them. Identified conflicts would then be resolved based upon conflict resolution procedures as required by the model's objectives. This is similar to the approach associated with time-driven simulation.

Unfortunately, such a simple conceptual strategy requires extensive computational resources. Moreover, computational resource demand grows with at least the square of the number of objects in the model. What is sought is an approach which enables conflict recognition between objects while minimizing the number of communication channels and interchanges which must be maintain between objects.

One proposed strategy currently under consideration in the author's research is based on a spatial blackboard. Under this concept, all objects of the object model are represented as geometric shapes in a so-called spatial blackboard. To simplify, all object model shapes are polygons. Static model-object (e.g.,

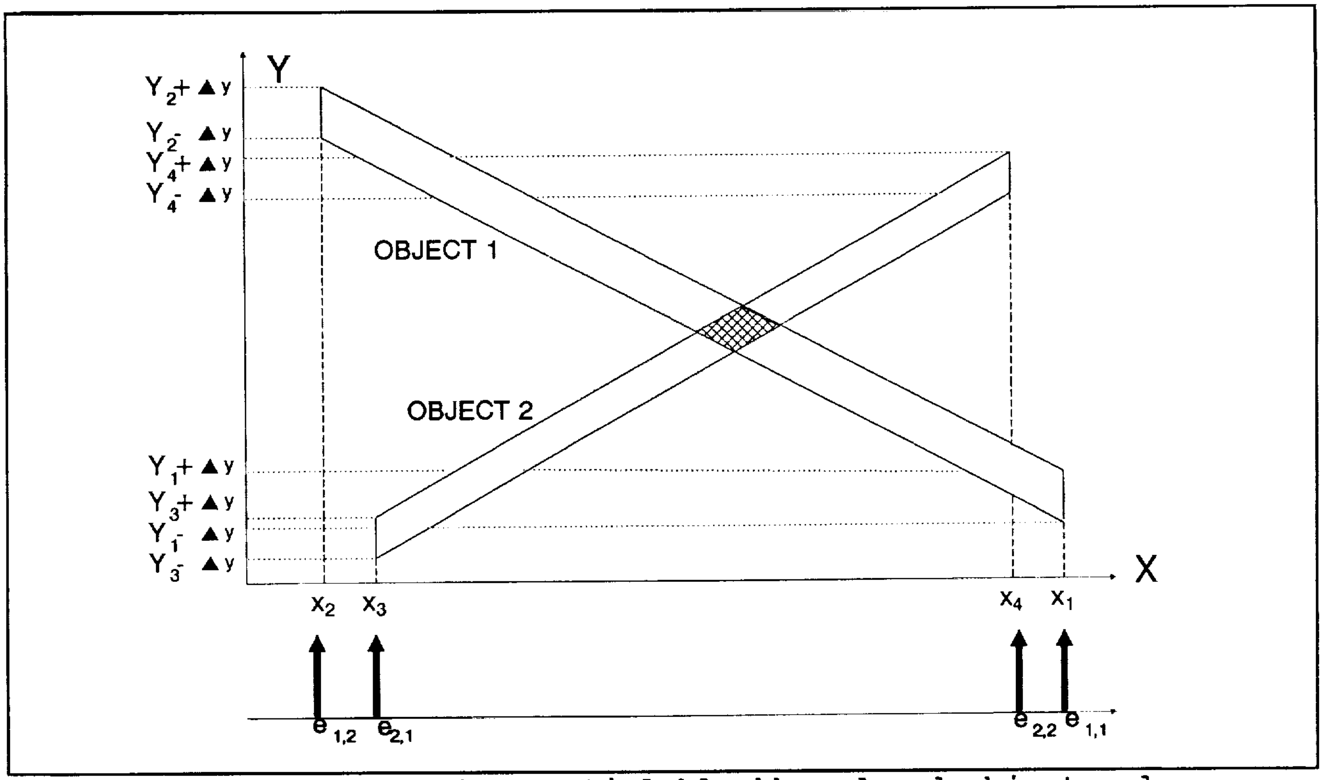


Figure 4. An example of a spatial blackboard and object polygons.

buildings, mountains, etc.) dimensions are approximated by a polygon. Dynamic objects (e.g. airplanes, ships, guided vehicles, weather, etc.) are also represented by polygons. The parameters of the associated polygon of the dynamic objects in the spatial blackboard are defined by the originating object event and the next scheduled event for that object as well as unique characteristics of the object.

For example, the trajectories of the two objects of the previous discussions could be represented as two polygons in the X-Y plane of their movement. The origin of Object 1's trajectory is point X_1,Y_1 and the end of its trajectory is point X_2,Y_2. These points correspond to events $\mathbf{e}_{1,1}$ and $\mathbf{e}_{1,2}$. Similarly for Object 2, its origin and end points are X_3,Y_3 and X_4,Y_4, respectively. The corresponding events are $\mathbf{e}_{2,1}$ and $\mathbf{e}_{2,2}$. The polygon on spatial blackboard for each of these objects could be defined by the origin and end points and by a Δy for each object. Thus, a polygon representative of Object 1 would be defined by the four points $(X_1,Y_1+\Delta y)$, $(X_1,Y_1-\Delta y)$, $(X_2,Y_2+\Delta y)$, and $(X_2,Y_2-\Delta y)$. A polygon representation of Object 2 would be defined by the four points $(X_3,Y_3+\Delta y)$, $(X_3,Y_3-\Delta y)$, $(X_4,Y_4+\Delta y)$, and $(X_4,Y_4-\Delta y)$. The two corresponding polygons and spatial blackboard are illustrated in Figure 4. This static representation in two dimensions implies a third dimension (time) by the extent of the polygon from point of the arrival event. This static representation may also be regarded as a most probable state trajectory for the object. Uncertainty in the proposed state trajectory may be reflected in the shape and extent of the object polygon (e.g. the Δys). Figure 4 illustrates this concept. Uncertainty as a function of time in the proposed state trajectory can be reflected by different Δys for the polygon points associated with the scheduled events $\mathbf{e}_{1,2}$ and $\mathbf{e}_{2,2}$. An object's polygon is reevaluated only when it completes its next scheduled event or the need to resolve potential conflicts necessitates establishing a new next scheduled event.

Potential conflicts are identified when the representational polygons of objects intersect. Once intersection is detected, objects may then go to a hierarchy of strategies for identifying and resolving conflicts. One strategy might be to reschedule an object's next event such that no potential conflicts occur between the t_{now} and the next scheduled event. The logic would be that by not projecting so far into the future (e.g. the length of object polygon) that, while a potential conflict was identified, the new possible state trajectories referenced from the later common time t_{now} would indicate that no actual conflict would have occurred in the trajectory of the two objects. If the proximity of the objects is such that the potential conflict does, in fact, exist, then the objects must respond by instituting some action to eliminate the conflict. This may involve simple decision rules, communication between objects or a third object, or higher ordered rule-based inferencing.

Clearly, in the spatial blackboard approach, two key methodological issues are central. One, how to efficiently identify polygon intersections in the spatial blackboard. Two, how to determine and select the next scheduled event for a dynamic object.

INTERSECTION IDENTIFICATION

Polygon intersection identification is essentially a graphical based methodology. The problem of how to determine if any two objects intersect in a three-dimensional graphical representation by the computer is a well known and widely addressed problem. Efficient and easily implementable methodologies are readily available. The task with polygon intersection detection that is most open at this time is what is the most efficient way to determine which objects should be tested to determine if they intersect. Testing between all objects in the system every time a new event is scheduled appears clearly undesirable.

Two further issues follow from these considerations. One, what data structure is most appropriated for representing the spatial blackboard. Two, what search approach should be used to identify objects for intersection evaluation. These two issues are related. One is needed to support the other and vice-versa. There is a great deal of literature on spatial data structures associated with database and computer graphics technology. Likewise, there is large body of information on computer search strategies. Work is currently focusing on identifying an appropriate combination of data structure and search strategy for the project.

EVENT SCHEDULING

Two potential strategies are under investigation for determining how state changes (i.e. events) are scheduled by an object. The first approach is *appointment scheduling*. With appointment scheduling, under normal operations each object employs a specified increment of time to schedule its next state change. After each time increment, the object determines if any other object have intersected its envelope. If an intersection has taken place, the object contacts the other objects whose envelopes have intersected. The extent of the potential conflict is assessed. The response may be to continue with smaller time increment and new envelope geometry. Or, the response may be to take some action based on the operational procedures associated with an object. In either case, the action taken by an object will depend on its own unique characteristics and its current state. The appointment scheduling approach may be considered as a conservative approach. The object schedules forward in time to the extent that would permit recognition of conflicts before they are scheduled. The time increment may obviously depend on the object's state at any given time.

The second approach is *goal scheduling*. The goal scheduling approach schedules the next event based upon some specified state goal (e.g., some destination point in the airspace model). A specific shape geometry is selected to identify a spatial envelope between the current location and the goal of the object. If a potential conflict is detected by the intersection of two or more envelopes, the goals of the affected objects are redefined to a goal where no conflicts are present. If no new goal of the original object objective can be established without a potential conflict, the affected objects are contacted. The conflict procedure then is resolved based upon the operational procedures of the objects or some control object. Goal scheduling may be viewed as optimistic scheduling. The object schedules its next event as far in advance as possible to obtain its final objective. It modifies this goal only after the criteria of potential conflict has been established.

Goal and Appointment scheduling are not mutually exclusive. There is nothing in the logic of these approaches which would prohibit a different objects employing different event scheduling approaches. Individual objects type might find one or the other to be more advantageous. Further, a given object might employ a combination of event scheduling methods depending on the circumstance.

It should be noted that the strategies of appointment and goal scheduling may appear analogous to conservative and optimistic approaches to parallel discrete event simulation. The approaches of parallel discrete event simulation reflect attempts to prevent deadlock and to manage efficiently the execution of a simulation distributed over multiple processors. The approaches properly reflect specialized operating systems of the computer resources. The strategies of appointment and goal scheduling are not concerned with management and utilization of the physical computing resources. The strategies of goal and appointment scheduling represent an approach that is a part of the simulation modeling paradigm and reflects approaches to the management of the interaction of objects within the model.

SUMMARY

The availability of autonomous object in event-driven discrete event simulation modeling would provide the capability to investigate new problems and incorporate levels of complexity not readily address in current simulation modeling paradigms. The major problems development of such use of autonomous objects are associated with synchronization of dynamic objects. The spatial blackboard offers one approach to this synchronization if viable approaches to the issues surrounding polygon intersection identification and event scheduling can be established. Efforts are ongoing is exploring this problems

ACKNOWLEDGEMENT

This work was supported by NASA-AMES Grant NAG-2-625.

SIMULATION REQUIREMENTS

- FOUR DIMENSIONAL MODELING IN DISCRETE EVENT SIMULATION
- THREE DEGREES OF SPATIAL FREEDOM FOR SIMULATION OBJECTS
- SPATIALLY BASED DECISION RULES AND PROCESSES
- SPATIALLY BASED PERFORMANCE MEASURES
- COMPUTATIONAL EFFICIENCY

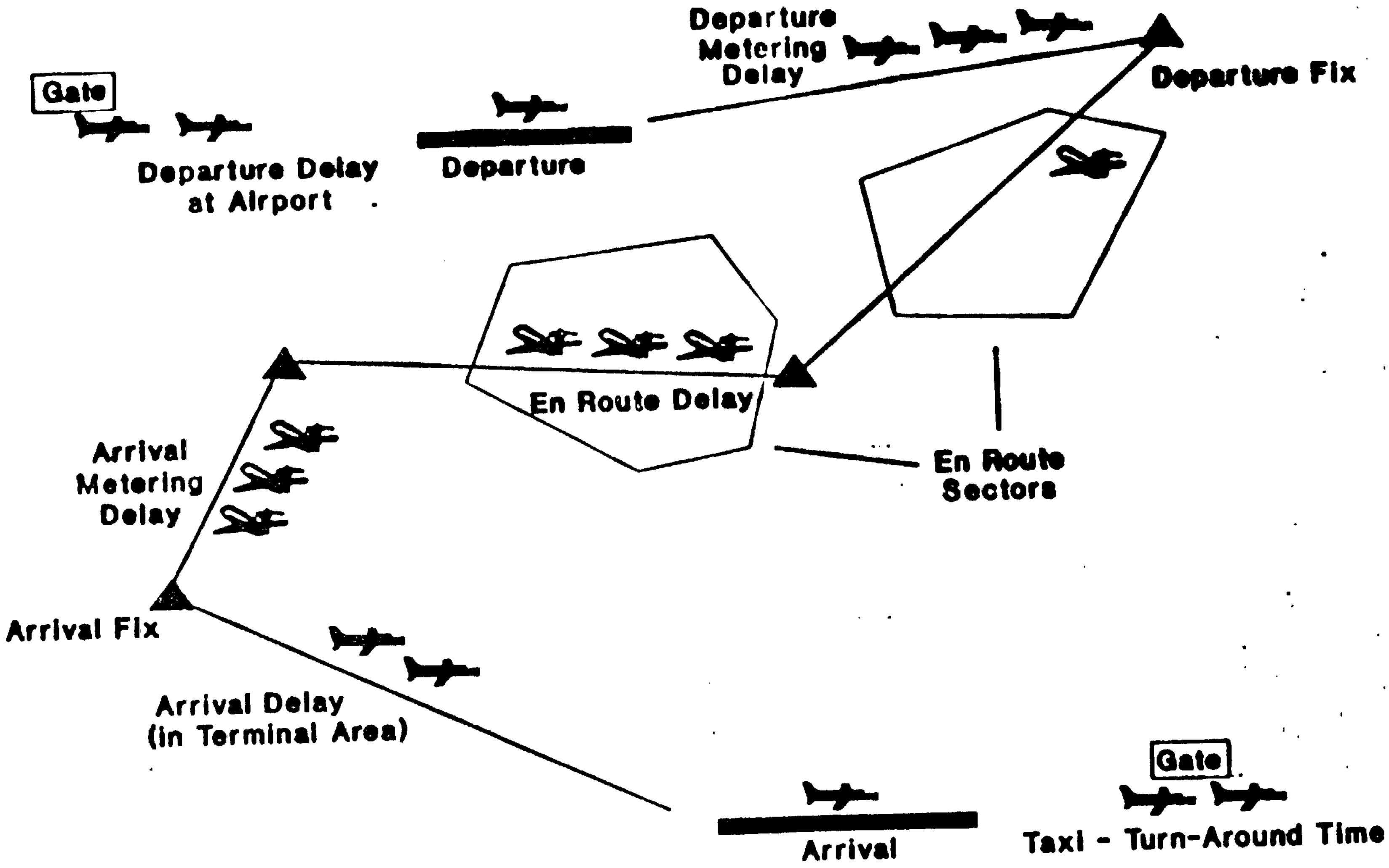
TIME DRIVEN

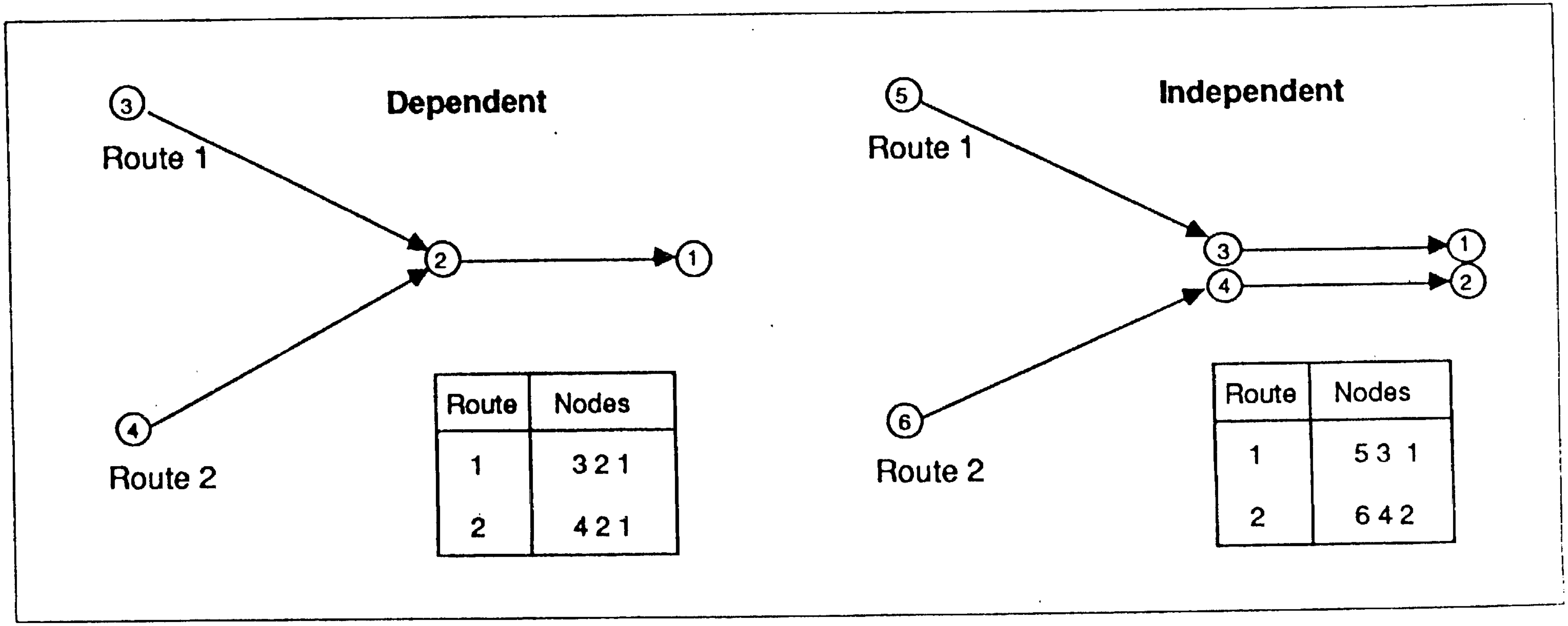
- **FIXED INCREMENT TIME ADVANCE**
- **AT INTERVAL START**
  - o IDENTIFY CONFLICTS
  - o OPERATIONAL DECISION

EVENT DRIVEN

- TIME ADVANCED BY EVENTS
- AT FIXED POINTS
  - IDENTIFY CONFLICTS
  - OPERATIONAL DECISION
- TWO DIMENSIONAL NODE AND ARC MODELS

SIMULATION MODEL
SEQUENCE OF EVENTS AND DELAY MEASUREMENT



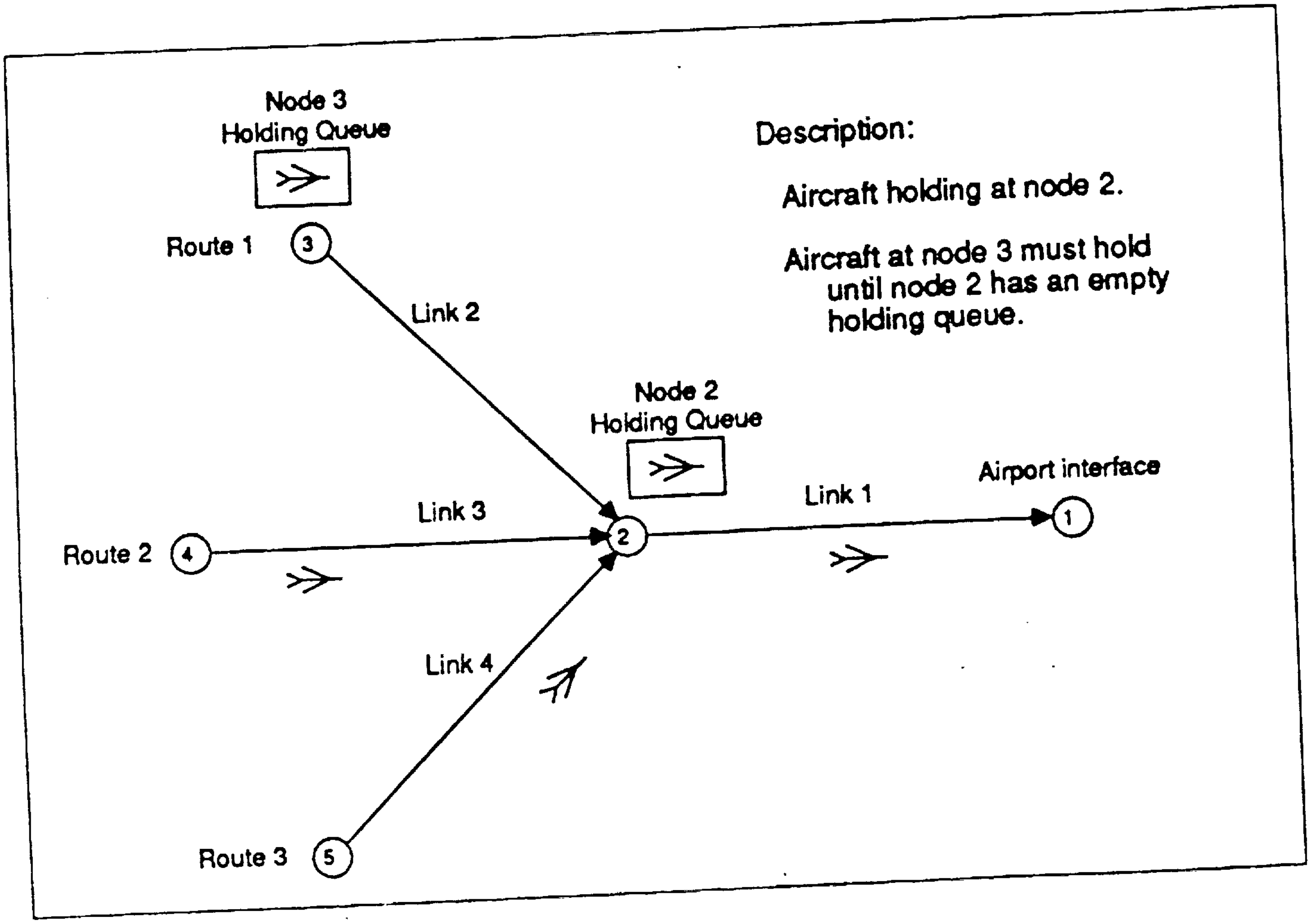


Dependent

Route	Nodes
1	3 2 1
2	4 2 1

Independent

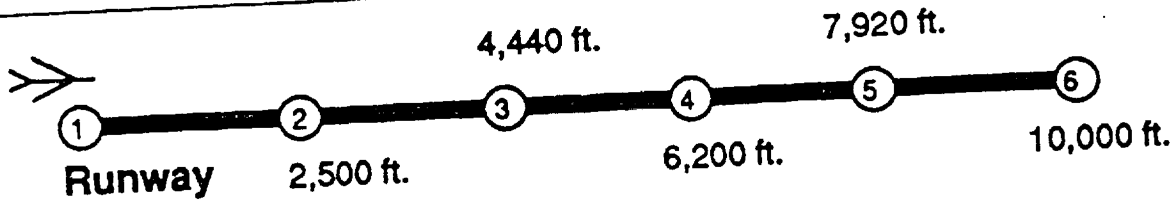
Route	Nodes
1	5 3 1
2	6 4 2



Description:

Aircraft holding at node 2.

Aircraft at node 3 must hold until node 2 has an empty holding queue.



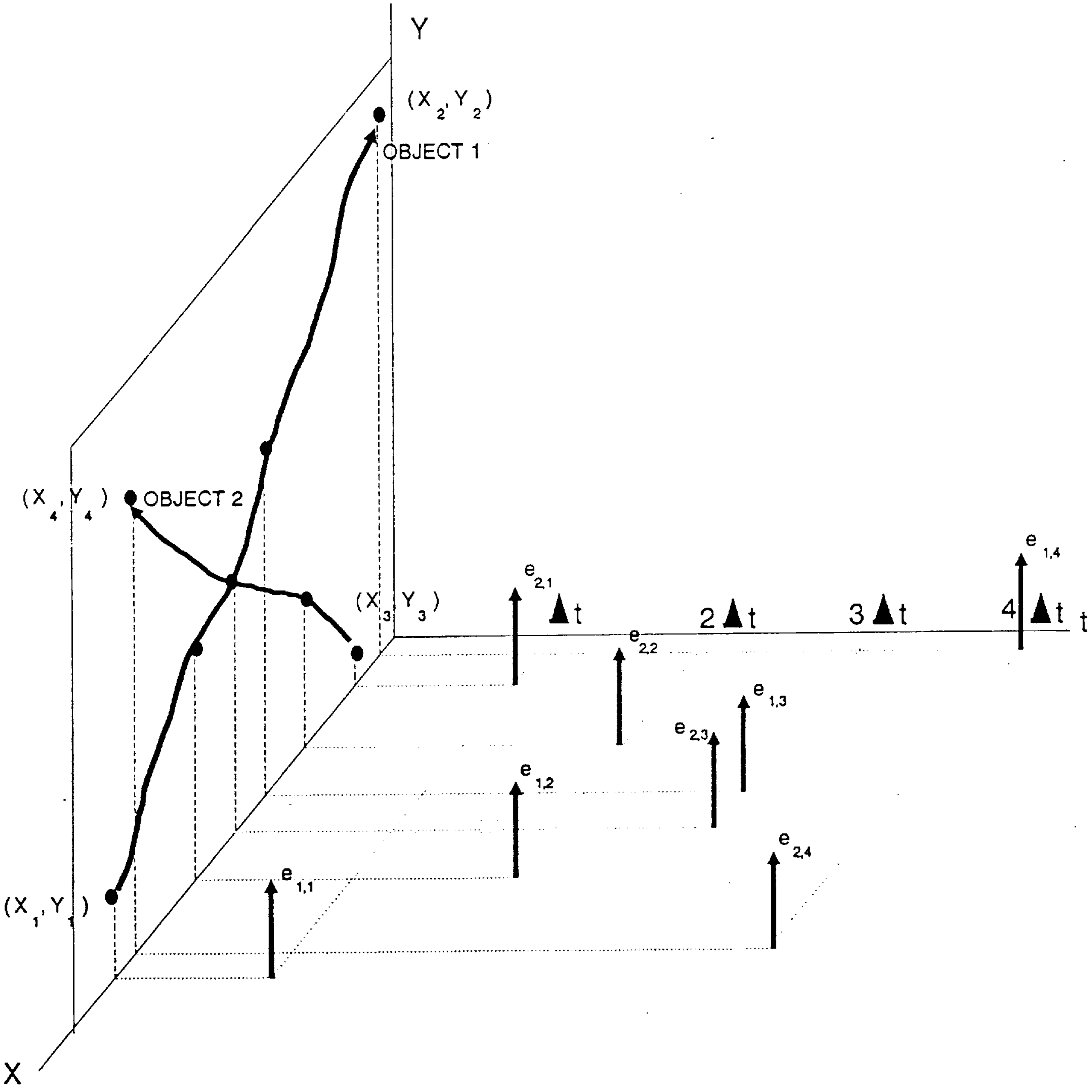
Roll distance of 6,200 ft
Roll time 50 seconds

Node	Cumulative Roll Distance	TOA at node
1	0	1:50:00
2	2,500	1:50:20
3	4,440	1:50:36
4	6,200	1:50:50

TIME-DRIVEN SIMULATION

Synchronizes object activity at each time increment

Positional precision function of time increment

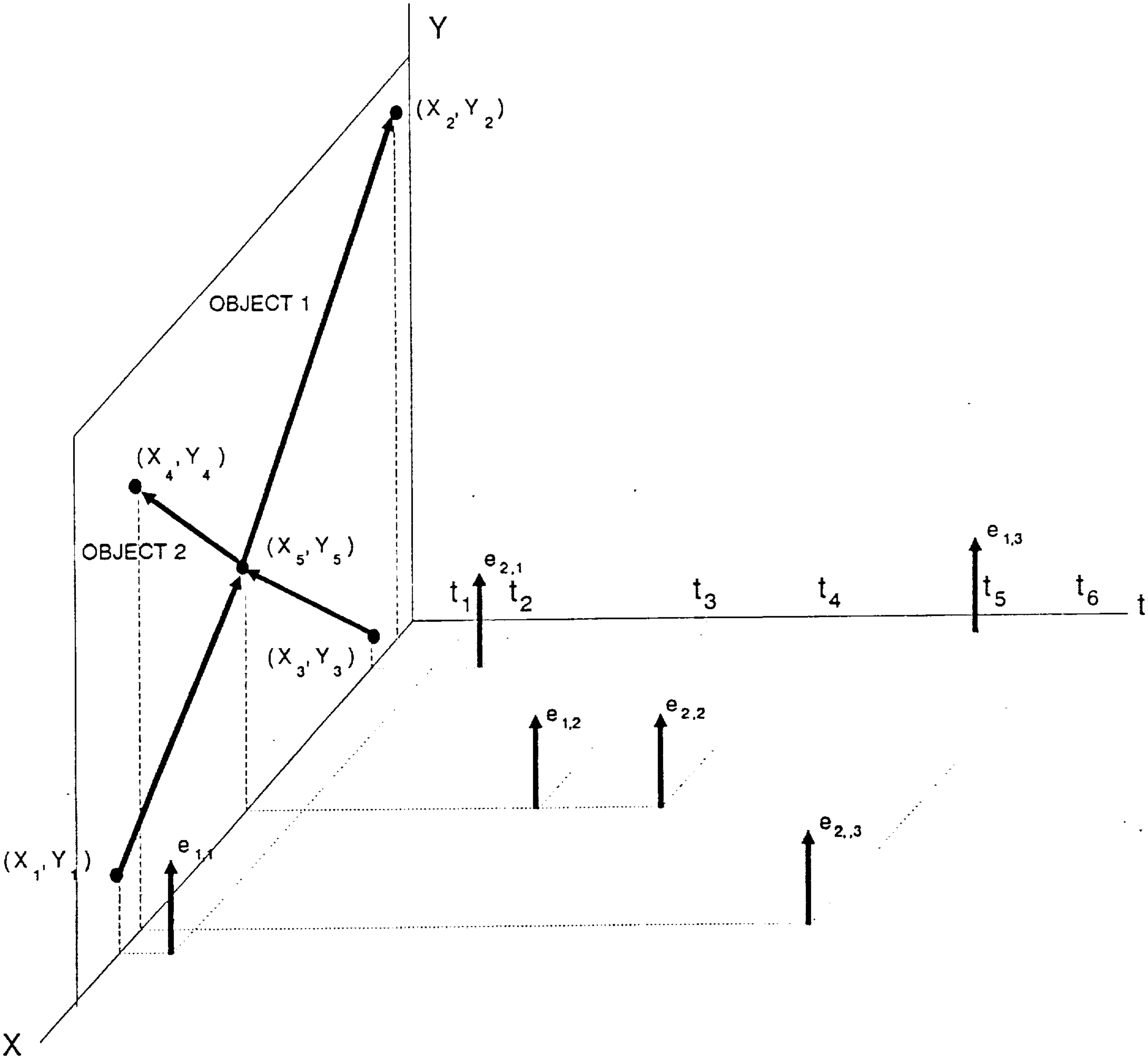


Y
(X_2, Y_2)
OBJECT 1
(X_4, Y_4)
OBJECT 2
(X_3, Y_3)
$e_{1,4}$
$e_{2,1}$
▲t
2 ▲t
3 ▲t
4 ▲t
t
$e_{2,2}$
$e_{1,3}$
$e_{2,3}$
$e_{1,2}$
$e_{2,4}$
$e_{1,1}$
(X_1, Y_1)
X

EVENT-DRIVEN SIMULATION

Synchronizes object activity at fixed reference points in Euclidean Space

Positional precision is defined by spacing between the fixed reference points



Y
(X2, Y2)
OBJECT 1
(X4, Y4)
OBJECT 2
(X5, Y5)
(X3, Y3)
(X1, Y1)
e1,1
e2,1
e1,2
e2,2
e2,3
e1,3
t1
t2
t3
t4
t5
t6
t
X

AUTONOMOUS OBJECTS

Objects which can alter their temporal and spatial goal trajectories based on current system state.

THE PROBLEM

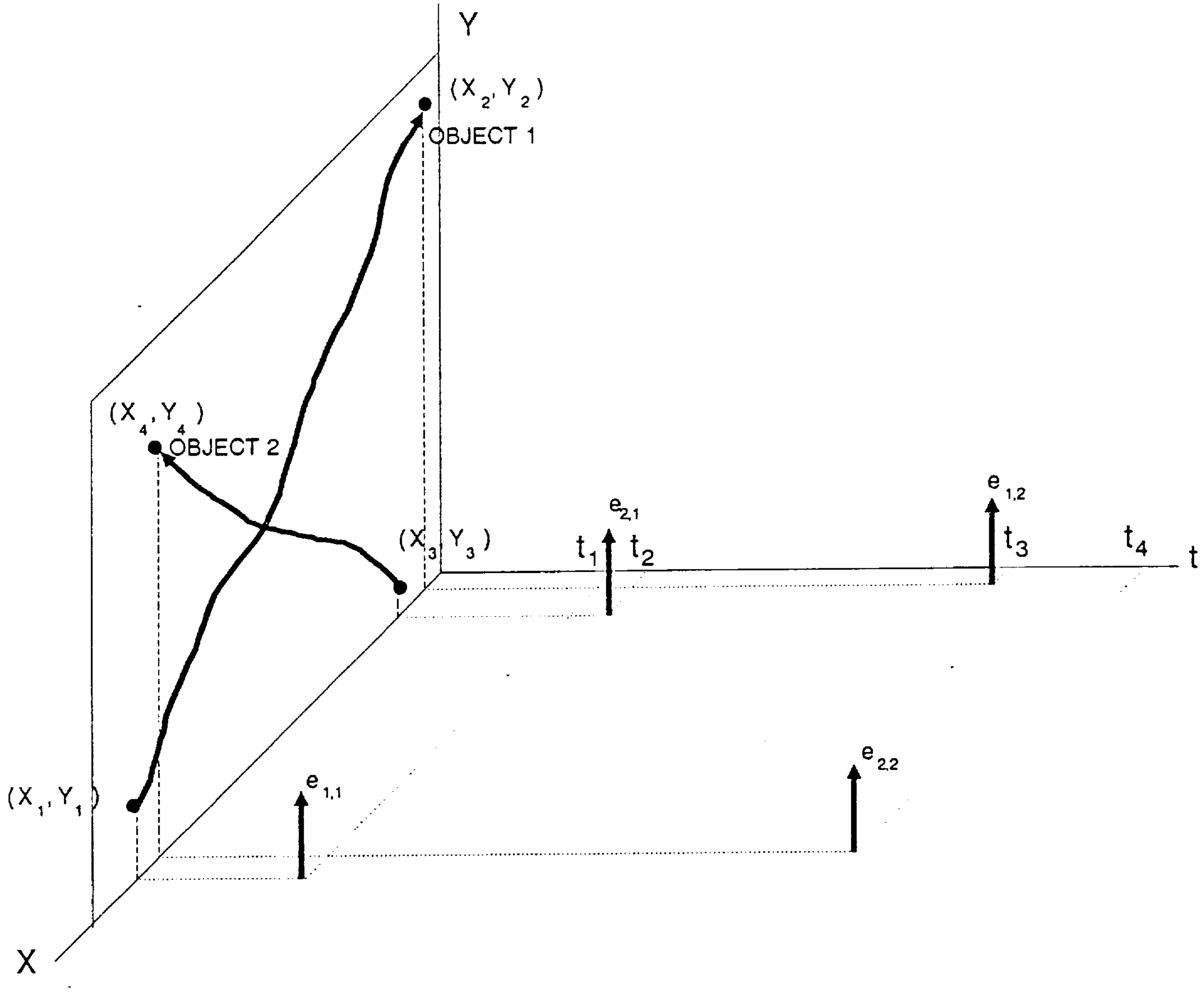
OBJECT SYNCHRONIZATION

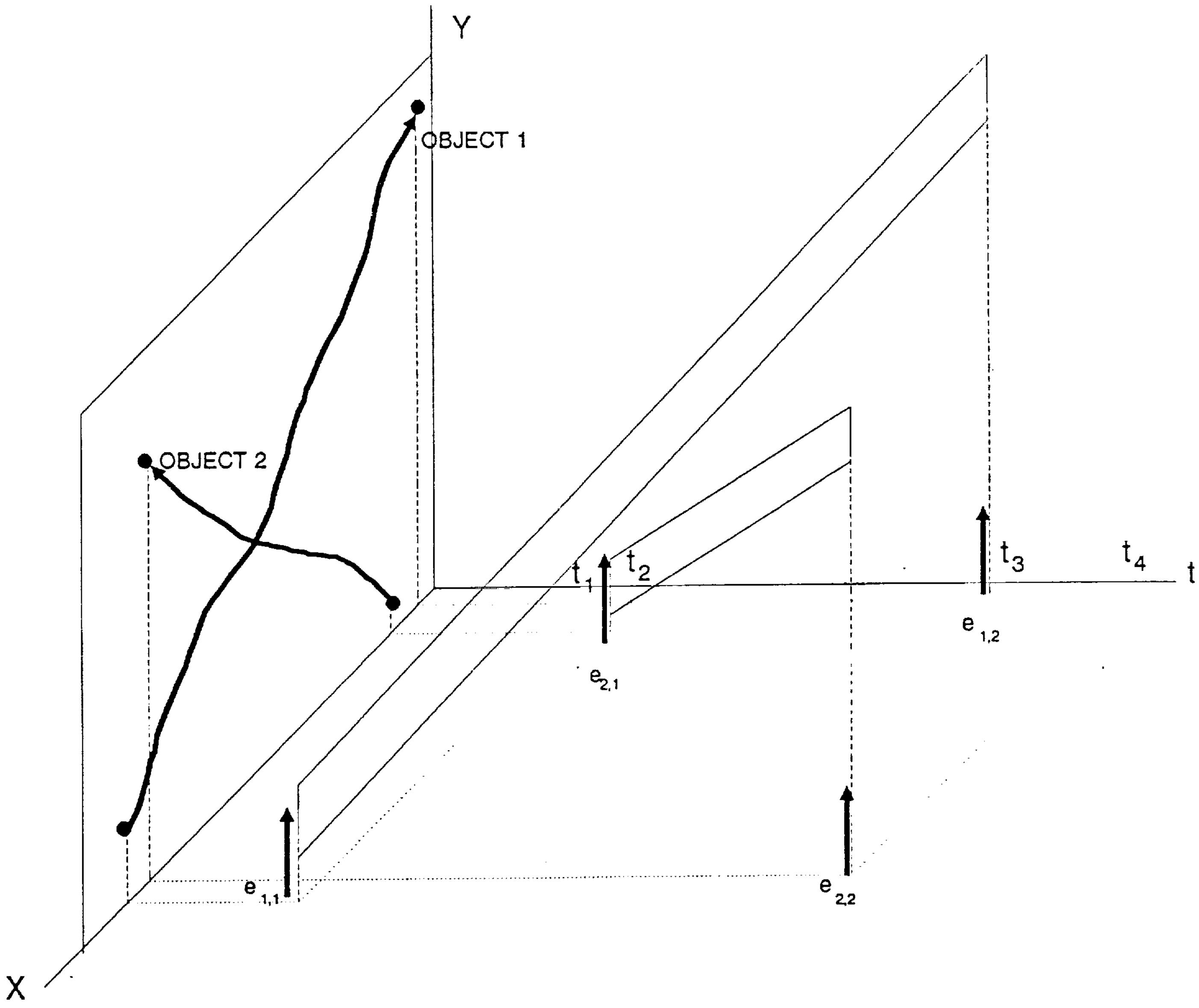
AUTONOMOUS OBJECT PARADOX

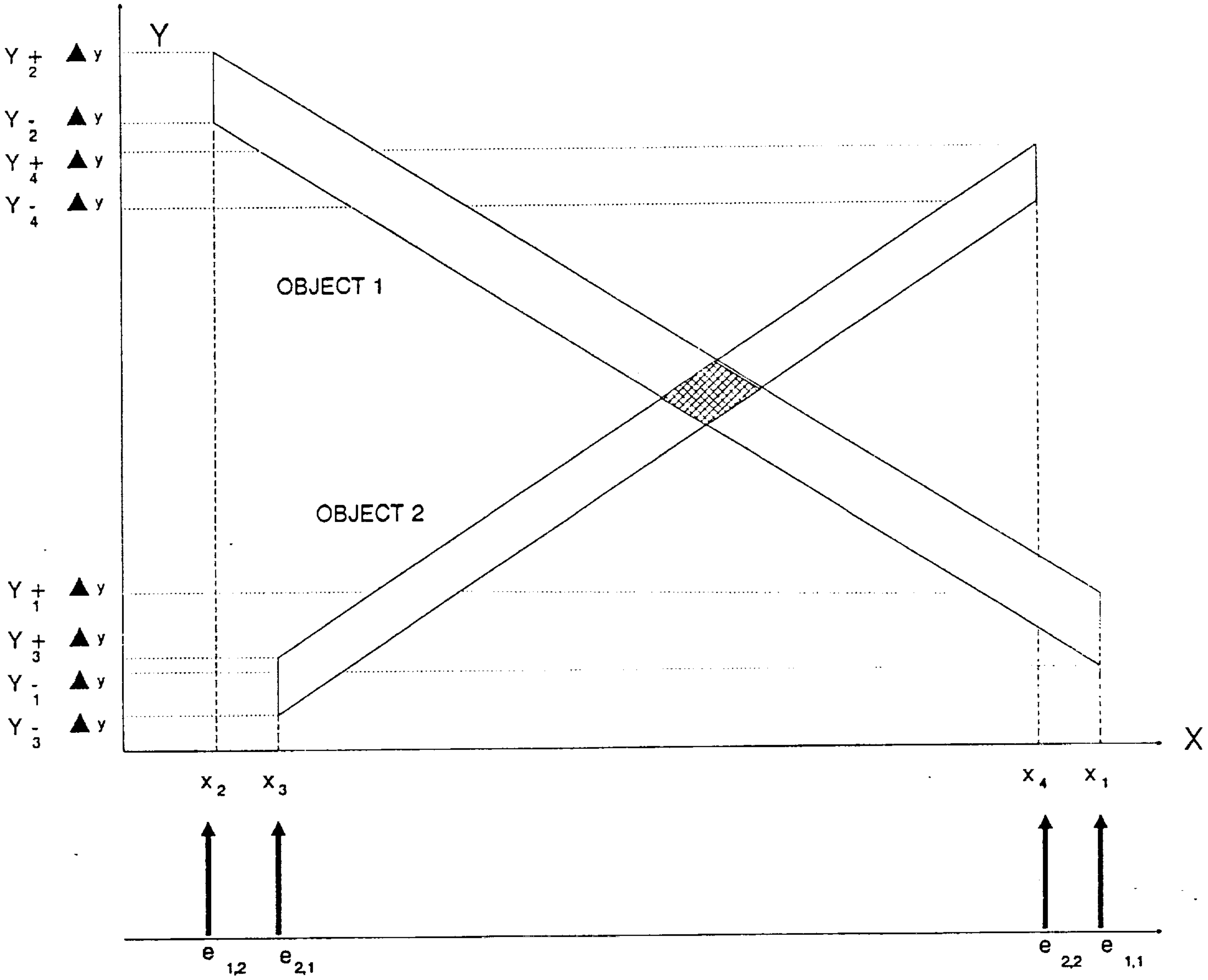
- Autonomous objects have freedom to modify trajectory in 3 space as goals and conditions require.

- To synchronize activity among objects some common reference space must be established.

- If an object must be tied to a common reference point, it is not autonomous.


Y
(X₂, Y₂)
OBJECT 1
(X₄, Y₄)
OBJECT 2
(X₃, Y₃)
e2,1
t1
t2
e1,2
t3
t4
t
(X₁, Y₁)
e1,1
e2,2
X


Y
OBJECT 1
OBJECT 2
t
t_1
t_2
t_3
t_4
$e_{1,2}$
$e_{2,1}$
$e_{1,1}$
$e_{2,2}$
X


Y
$Y_2 + \Delta y$
$Y_2 - \Delta y$
$Y_4 + \Delta y$
$Y_4 - \Delta y$
OBJECT 1
OBJECT 2
$Y_1 + \Delta y$
$Y_3 + \Delta y$
$Y_1 - \Delta y$
$Y_3 - \Delta y$
X
x_2
x_3
x_4
x_1
$e_{1,2}$
$e_{2,1}$
$e_{2,2}$
$e_{1,1}$

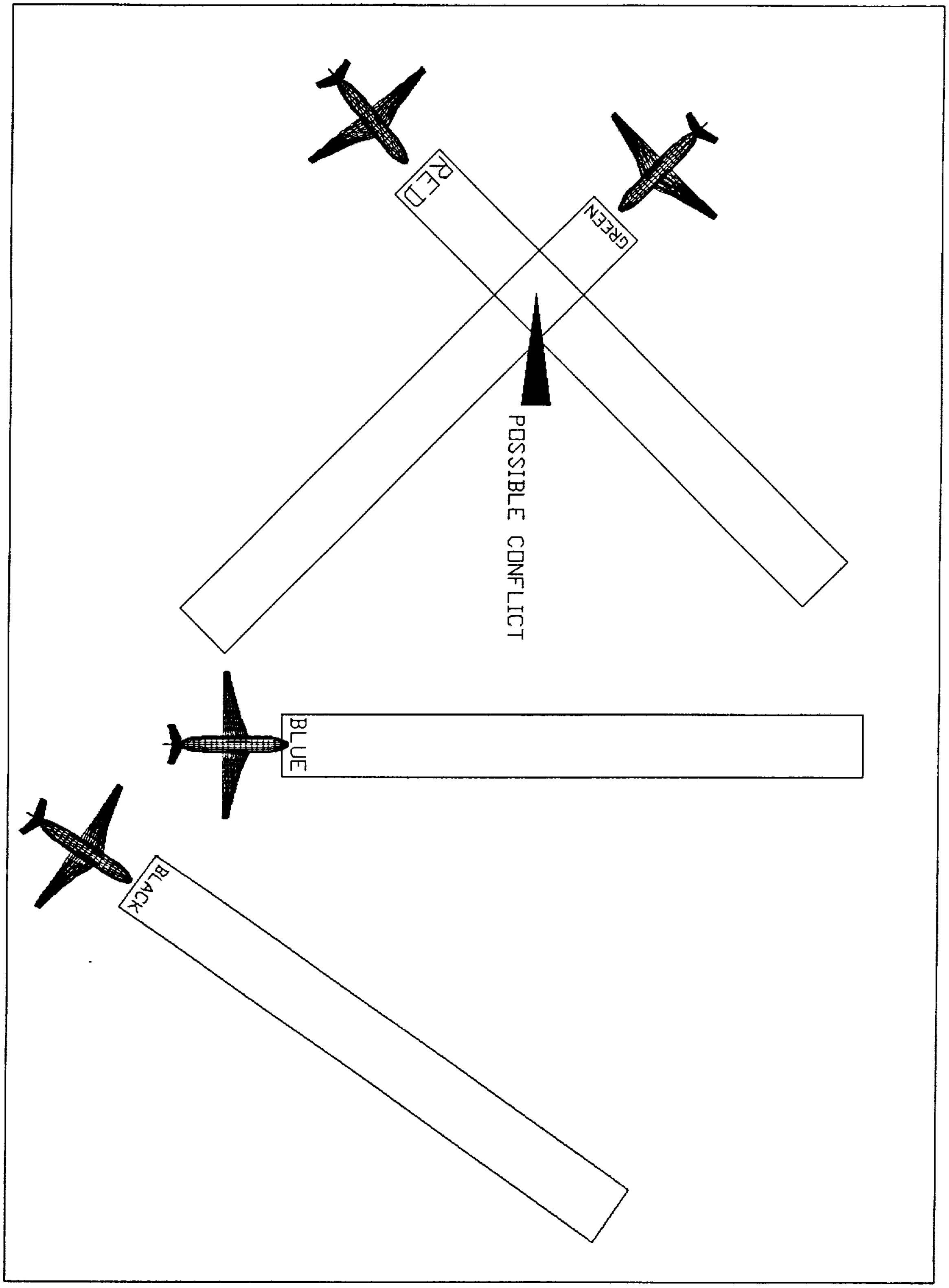
THE NEXT PROBLEMS

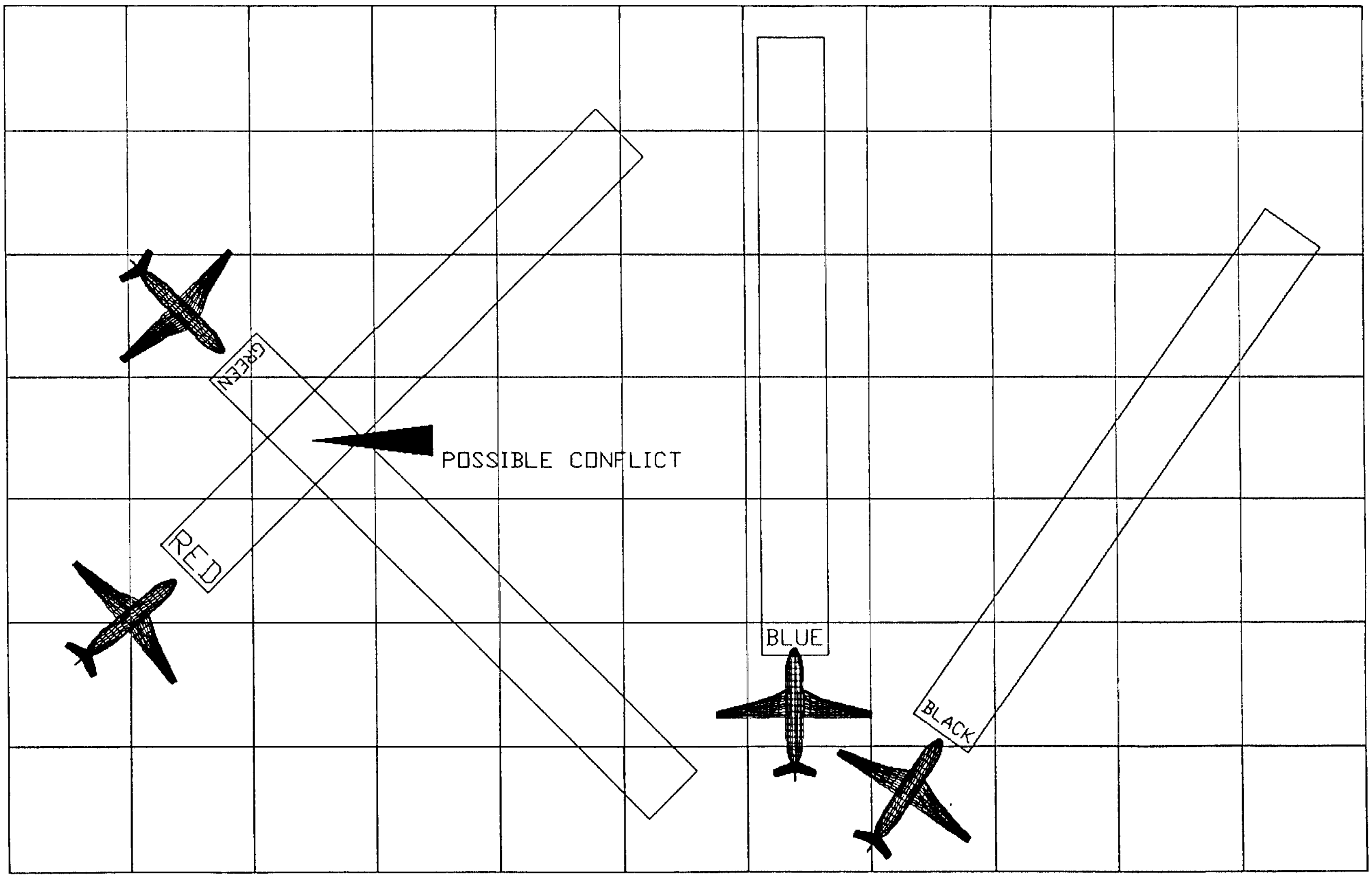
- Recognizing and communicating potential conflicts

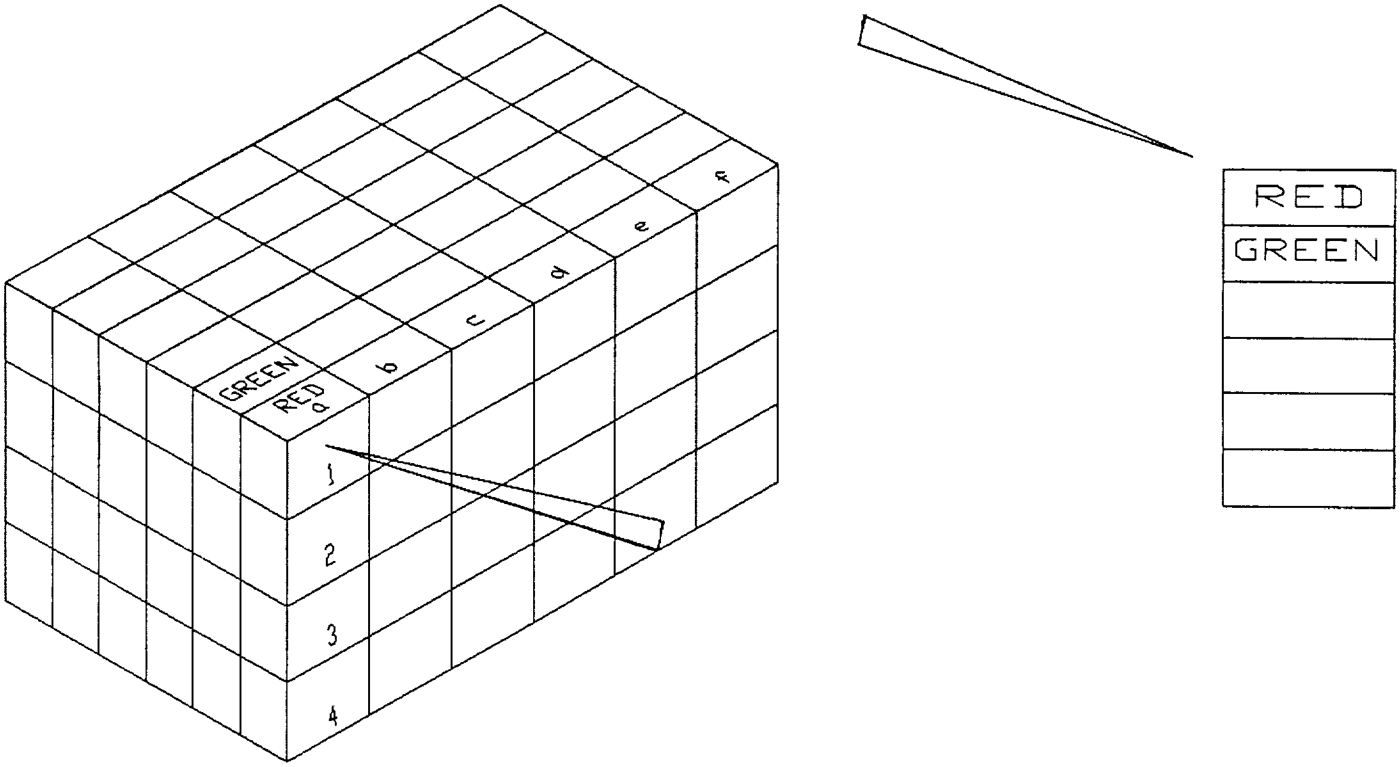
- Scheduling next state change (i.e. event)

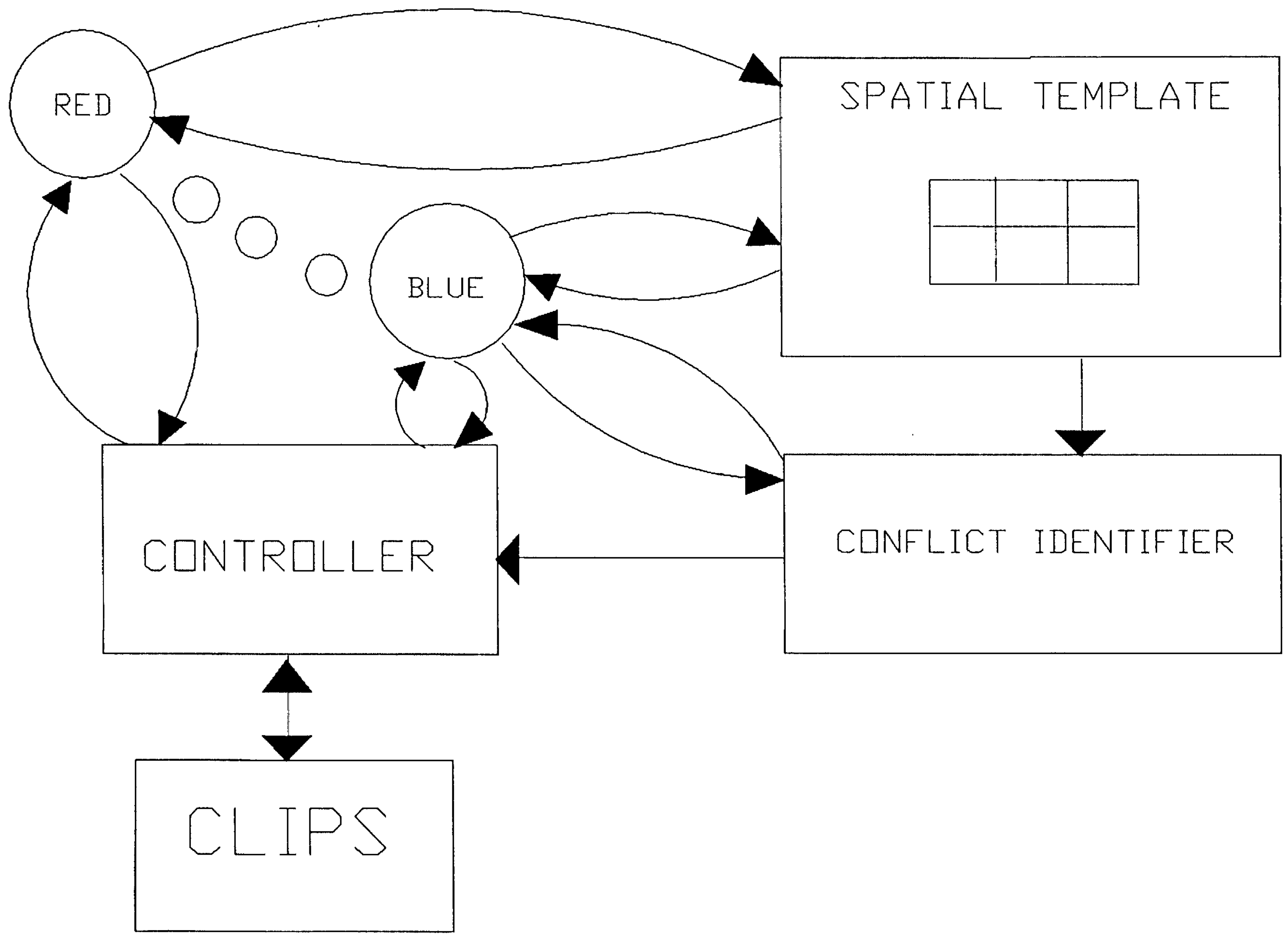
CONFLICT RECOGNITION

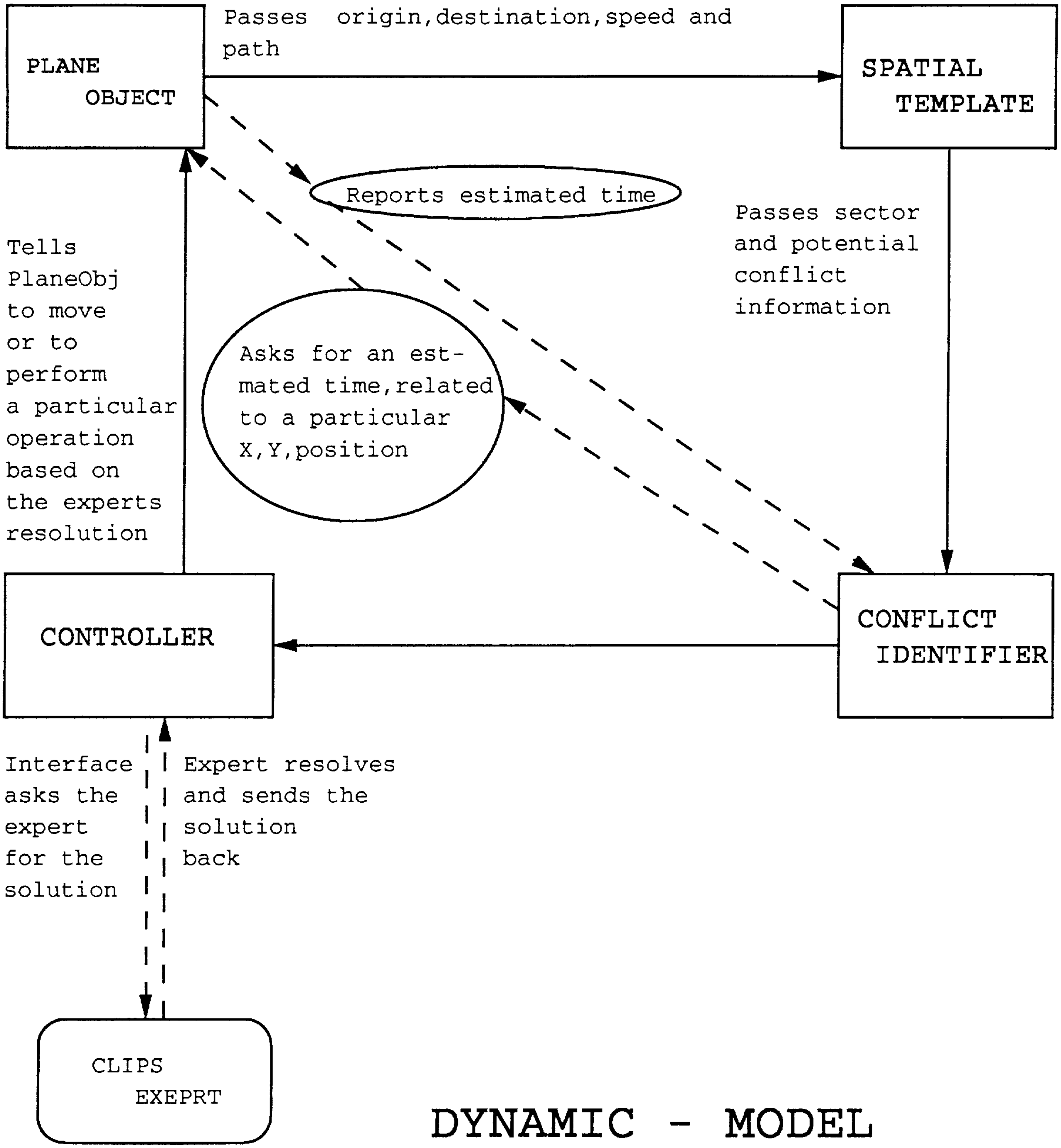
- Spatial Data Structure
- Search Strategy


RED
GREEN
POSSIBLE CONFLICT
BLUE
BLACK


GREEN
POSSIBLE CONFLICT
RED
BLUE
BLACK


GREEN
RED
a
b
c
d
e
f
1
2
3
4
RED
GREEN


RED
SPATIAL TEMPLATE
BLUE
CONTROLLER
CONFLICT IDENTIFIER
CLIPS


PLANE OBJECT
Passes origin,destination,speed and path
SPATIAL TEMPLATE
Reports estimated time
Passes sector and potential conflict information
Tells PlaneObj to move or to perform a particular operation based on the experts resolution
Asks for an estmated time,related to a particular X,Y,position
CONTROLLER
CONFLICT IDENTIFIER
Interface asks the expert for the solution
Expert resolves and sends the solution back
CLIPS EXEPRT

DYNAMIC - MODEL

(EVENT FLOW DIAGRAM)

SIMMOD

- NODE AND LINK MODELS
- FUNDEMENTALLY TWO DIMENSIONAL
  - ○ THIRD DIMENSION INCORPORATED AS ATTRIBUTE OF AIRCRAFT ENTITY
- CONFLICTS IDENTIFIED AND RESOLVED AT NODES
- SPATIAL RESOLUTION LIMITED TO NODE SPACING OF MODEL SPACE
- NO SPATIAL RANDOMNESS AVAILABLE ON LINK, ONLY TIME
- ALL POSSIBLE FLIGHT TRAJECTORIES MUST BE MODELED WITH NODES AND LINKS
- ALL SITUTATIONS MUST BE RECOGNIZED AT NODES AND RESOLVED AT NODES.
- CHANGES IN ALTITUDE ARE MADE BETWEEN NODES. CONFLICTS DUE TO CHANGING ALTITUDES NOT EASILY HANDLED
- SPATIAL MEASURES LIMITED TO CONFLICTED IDENTIFICATION AT NODES

- EMPHASIS ON TEMPORAL OR TIME BASED MEASURES SUCH AS CAPACITIES, DELAYS, NUMBERS OF TAKE-OFF AND LANDINGS

- DOES NOT EASILY ALLOW INCORPORATION OF SENSOR AND NAVIGATION TECHNOLOGY MODELS INTO SIMULATION MODEL

- MODELS ARE BRITTLE. THEY ARE NOT EASILY MODIFIED TO CHANGING SITUATIONS

- PROCEDURAL RULES MUST BE PART OF BASIC MODEL

- DOES NOT INTERFACE WITH EXPERT SYSTEM SHELL

- BASED IN TRADITIONAL DISCRETE EVENT SIMULATION TECHNOLOGY (I.E. SIMSCRIPT)