2011 Fourth International Symposium on Computational Intelligence and Design

Team Tactics in Military Serious Game

Jun Lai, Wei Tang, Yihui He
PLA University of Science and Technology,
Nanjing 210007, China
e-mail:luxboy2003@163.com

***Abstract*—Serious games[1] are widely used in military training. There are many game modes, the team command mode is more effective to enhance tactical skills. In this paper we discuss three priority Problems:Tactical Formation,Tactical Position and Tactical Path-finding. Finally, This approach to establish so-called team tactics, is tested in the UDK environment[2]. From our results, we may conclude that the approach to established Team Tactics can be successfully applied in actual military serious game.**

Keywords- AI; Team; Tactics; Serious Game; UDK

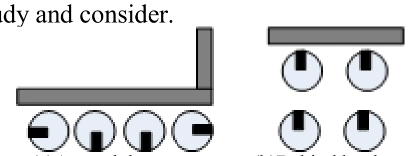
I. INTRODUCTION

In recent years, with the "hot games" in foreign military education and training in the heats, Computer games in training and upgrading the quality and ability of military personnel has revealed more widespread effect, has played an increasingly important role.

"Full Spectrum Warrior" is one of a typical training for actual combat game guide[3]. This game was originally used in the military for leadership training as well as decision making training. The main feature of the game is its team AI. Unlike first-person shooter games, players pay more attention to entertainment. They often ignores many tactical details, only care about the number of enemy killed. But he team command mode focus on the teams, the team tactics is the player most in need of attention, players only need to consider the tactics and commands issued. Next we come to discuss the realization of team tactics.

II. TACTICAL FORMATION

Current approaches to organising units in strategic video games are typically implemented via static formations.A formation is defined as an arrangement or disposition Static formations are not capable of adapting effectively to of units[4]. Formations are typically applied for in a particular terrain. But we don't need to consider the formation of the hold in the process of moving the team, the purpose is to quickly move through to the next defensive position, and the tactics formation in the defensive position is that we really need to study and consider.



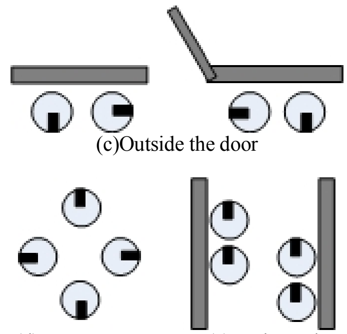


Figure 1. shape of the frmation

In Figure 1, the design of particular terrain formations are illustrated. Based on the multi-agent formation control method, we use the formation control method based on the Leader, a AI unit is designated as the Leader, the other as its Follower in the team, Follower to Leader certain distance interval tracking the location and direction. Under the Leader and the Follower relationship between the relative position, can form different network topologies, namely the formation of a different formation.

Taking into account the different position of the unit is different, units have different roles in different behavior, and follow the following principles:

(1) Leader choice the tactical position and determine the path to destination based on the current environmental information.

(2) Followers only remain the formation. At the same time, followers accept from the Leader of the relevant information, combined with their own specific circumstances of the current decision-making, and don't simply move according to Leader's instructions, but always aware of environmental information for emergency response.

Relationship between Leader and Follower position described below formula:

$$\begin{cases} x_{Follower} = x_{Leader} + x_{Team} \\ y_{Follower} = y_{Leader} + y_{Team} \end{cases}$$

III. TACTICAL POSITION

During combat the team will frequently change position to achieve its goal. For example, the team pick new positions to hide, obtain a better line of fire, close in with its target, or to obey a squad command. In general, the leader will attempt to find a suitable position in the direct surroundings of its current position, or in the area the team is commanding it to.

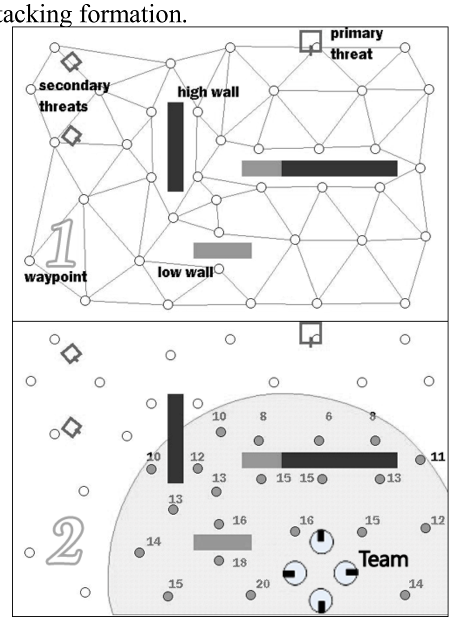
978-0-7695-4500-4/11 $26.00 © 2011 IEEE
DOI 10.1109/ISCID.2011.28

To select or pick a position to move to, the Leader considers all positions within a radius around its current position or area-of-operation's center. From these, the AI eliminates positions already claimed by others, as well as the positions outside the area-of-operations. For each of the remaining positions the position evaluation function is invoked, and the position is annotated with the resulting score. Higher scores suggest more attractive positions. The annotated positions are then sorted by score.

A post-processing step is performed to verify that the highest scoring position is suitable for the intended purpose. For example, when picking a cover in the center of a open area, even the highest scoring position will not actually provide cover. This will be obvious from the highest score, provided that the 'cover' input was given a stronger weight than all of the other inputs together. In cases like this the leader may decide to temporarily abort its search for cover.

The algorithm to find the best tactical position include the following seven steps:

- Step 1:Initial situation with waypoint graph and legend, build connected graph, Set score for each waypoint is zero;
- Step 2:Selected nearby waypoints, annotated with proximity.
- Step 3:Annotations for positions with a line-of-fire to primary threat.
- Step 4:Annotations for positions with cover from the secondary threats,all add to the corresponding score;
- Step 5:Annotations for positions inside the preferred fighting range,add to the corresponding score;
- Step 6:Adding up all the annotations yields the best attack position. According to tactical formations, fine adjust the position and orientation, form the best attacking formation.



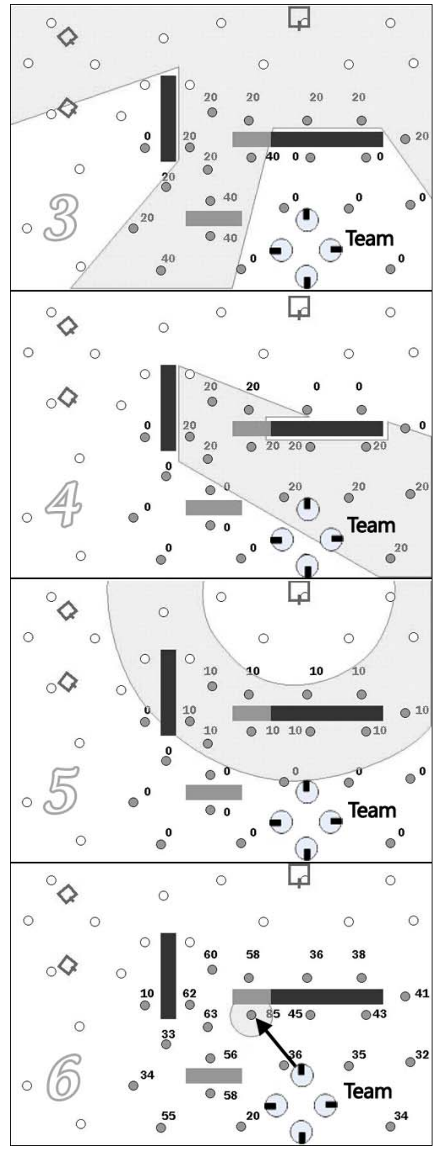


Figure 2. tactical position picking

The minimal world presentation necessary to support the tactical behavior described in this paper consists of:

- navigation info supporting position enumeration and connectivity queries
- visibility info supporting line-of-fire (LoF) and line-of-sight (LoS) queries.

A. Navigation, position claims and danger zones, and areas-of-operations

The navigation system has to be capable of generating all accessible locations near a given position, and of listing all locations in direct connection to a given accessible location. It does not matter whether the navigation system is based on meshes, on cells or on waypoints, as long as its granularity matches the size of the individual cover and attack positions.

For navigation purposes, we maintain a connectivity graph for the waypoints, where the links are annotated by estimated travel time. This is a lightweight representation, where it would be easy to adapt the structure on the fly in response of terrain changes.

The AI uses some additional information in its position picking and path-finding. All AI units can claim as their destination positions. Other friendly units then attempt to not to pick these claimed positions or move through them.

An AI unit also keeps track of (observed) entities that present actual dangers such as an incoming grenade. Danger zones are used to represent the area where danger exists.

B. Lines-of-sight,lines-of-fire and a 'worst-case' visibility lookup table

The serious games feature complex character models and world geometry. It also allows the player and AI characters to assume multiple postures (stand, crouch). As a result, it is not easy to answer whether one character is able to see (part of) another character. And it is certainly difficult to answer whether a character would be able to find cover at any of the nearby locations from multiple threats who are assuming postures offering them the best line-of-fire. However, the tactics we want to create in our AI heavily depend on these kinds of queries.

To efficiently answer the numerous line-of-sight and line-of-fire checks, we use a look-up table for waypoint visibility, for standing and crouched stances. To fit in a modest amount of memory, this table does not attempt to record accurate visibility for each pair of waypoints. Instead stores an approximation of that visibility in a polar representation*(Figure 5)*:

For each waypoint w_t, each stances, and each direction d in the polar representation, the table entry for (w_t, s, d) contains the largest distance for which an AI actor in stance s at or near waypoint w_t does not have cover from an attacker in some stance s' at or near some waypoint w_a positioned in direction d relative from w_t.

Using this table, an AI character positioned in stance s near a waypoint w_t is guaranteed to have cover from a ground based attacker at distance z in direction d if z is greater than the distance recorded for w_t, s and d . The reverse is not necessarily true: when a threat is closer than the recorded distance for w_t, s and d, this threat need not have a line-of-fire.

We assume symmetry for lines-of-fire. Solely when the table states that w_t does not have cover from w_a, and that wa does not have cover from w_t, there might be a line of fire from w_a to w_t. Based on this assumption, w_t is also guaranteed cover from w_a if the table states that wa has cover from w_t.

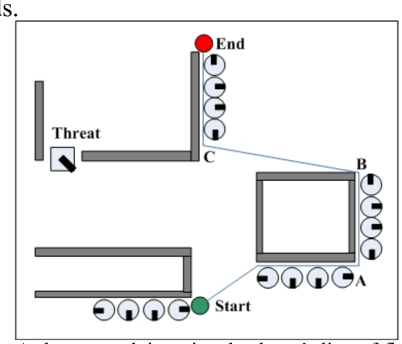
This table represents a "worst-case" assumption about cover for a given direction and distance: if there is a single position at a certain distance and in a certain direction from which an attacker can establish a line of fire in some stance,then it is assumed there is no cover from any position represented by the same distance and direction.

This "worst-case" cover assumption has two important benefits. First, although our table is inaccurate in its representation of line-of-fires, the AI can fully rely on a position offering cover when the table says so. Solely statements about a position not offering cover (thus having a line-of-fire) may be too optimistic and require verification with a ray cast. This is attractive, because in position picking and path-finding, the AI has a far greater need for reliable positive statements about cover than about lines-of-fire. For example, to pick an attack position the AI uses a line-of-fire query, but will also query for partial cover from the target and full cover from any secondary threats.

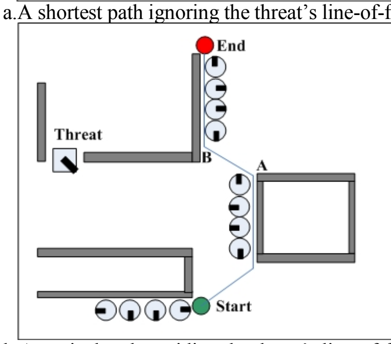
A second benefit of the "worst-case" cover assumption is a degree of robustness against threat movement and stance changes. Even when a threat effectively does not have a line-of-fire when the AI picks a position or plans a path, if that threat easily could establish a line-of-fire from a nearby position then our lookup table most likely states that the AI does not have guaranteed cover from the threat. The resulting robustness can be regarded as a limited ability to anticipate threat movement and stance changes.

IV. TACTICAL PATH-FINDING

Position evaluation functions can also be used to make AI paths more responsive and tactical. In cost functions for conventional (shortest-path) path-finding, the costs for a move from one position to another solely consist of the corresponding travel costs. Such a cost function can easily be extended to invoke a position evaluation function for each of the positions visited. This position evaluation function can, for example, add costs to make it more expensive (and less attractive) to cross the player's line-of-fire or to be visible to hostile guards.



a.A shortest path ignoring the threat's line-of-fire



b.A tactical path avoiding the threat's line-of-fire

Figure 3. tactical path-finding

In Figure 3 we see the effects of adding costs for visiting a threat's line-of-fire to the path-finder's cost function: the path takes a small detour to avoid the line-of-fire where possible.

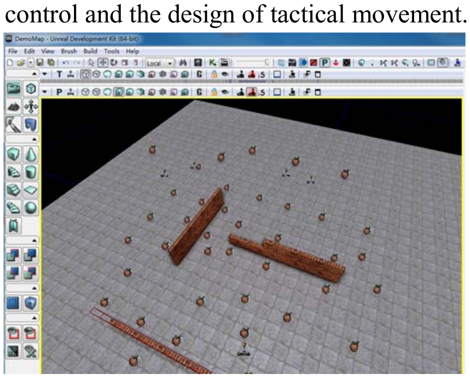
Although it is relatively straightforward to extend a path-finding algorithm with a position evaluation function, it then is more difficult to keep the run-time costs of that algorithm under control. The addition of position evaluation functions may cause the path-finder to explore a large part of the terrain.

we assign area-of-operations to each AI unit to control and limit where the units move, fight and hide. Consequently, we can have the A* path-finder restrict its search to the unit's area-of-operations. This keeps the path-finding efficient and the resulting paths predictable even in dynamic situations with several hostile and friendly lines-of-fire.

V. EXPERIMENT

To the above-mentioned team tactical thinking into a practical design and implementation of the military serious games, we used UDK as the development platform,UDK is Unreal Engine 3 – a complete professional development framework. We selected UTGame opponent model as the role of the robot, our team members used self-built model of the PLA, the team consists of 4 players.

Using the Unreal engine collision detection algorithm, the platform can automatically identify the walls, bunkers, and the corner, according to the previously mentioned tactical position picking and tactical formation control, in the experimental four players are able to successfully reach the intended destination. In the process of moving, the four units did not choose the short path, but choose the path through the tactics of the way quickly hidden to reach the destination in column, these experiments vertify the algorithm for formation control and the design of tactical movement.



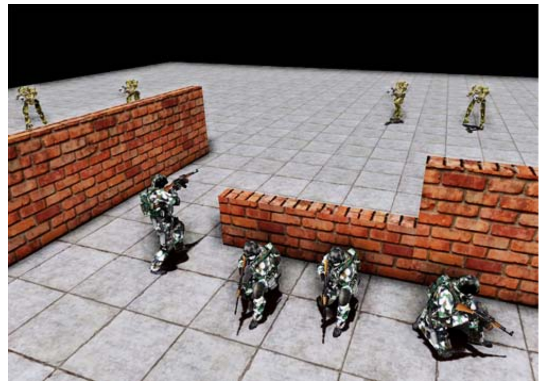


Figure 4. screenshot of the UDK game environment.

VI. CONCLUSIONS

Now, team tactics verification environment has been completed ,the AI units have extensive tactical action, which can show a good tactical behavior. The next step, we will continue to improve the team tactics, which can be applied in any terrain, and apply these tactics to the enemy intelligence, and improve system confrontational and practicality.

REFERENCES

[1] Antoinette Caird-Daley:Training decision making using serious games: Requirements analysis for decision making training(2009),P12

[2] Information on http://www.udk.com

[3] TngC.H.John, EdmondC.Prakash:Strategic Team AI Path Plans:Probabilistic Pathfinding(2007)

[4] Marcel van der Heijden, Sander Bakkes:Dynamic formations in real-time strategy games(2008)

[5] Li bo,Wang xiang feng:Dynamic Leader based multi-robot formation control(2009)

[6] Michiel Rook,Arno Kamphuis:Path Finding using Tactical Information(2005)

[7] R. Graham, H. McCabe, S. Sheridan:Neural Pathways for Real Time Dynamic Computer Games(2005)

[8] Darren Doherty and Colm O'Riordan:volving Agent-Based Team Tactics for ombative Computer Game(2006)

[9] Christian Thurau, Christian Bauckhage:Learning human-like Movement Behavior for Computer Games(2004)

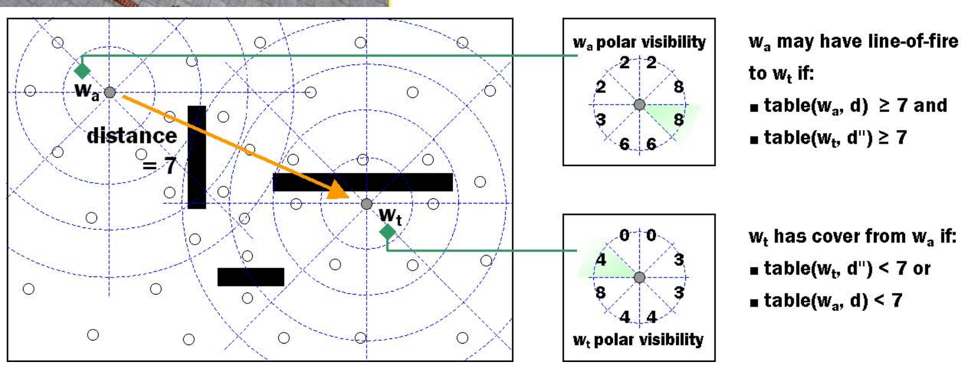


Figure 5. An example of the waypoint based polar visibility info and the line-of-fire and cover queries