

2

CONTRACTOR REPORT BRL-CR-641

BRL

AD-A226 322

SMALL FORCE EFFECTIVENESS
OF DIRECT FIRE WEAPONS:
MODEL ANALYSIS

HARRY L. REED, JR.
BATTELLE COLUMBUS
COLUMBUS, OHIO

SEPTEMBER 1990

DTIC
ELECTE
SEP 10 1990
S E D

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

U.S. ARMY LABORATORY COMMAND

BALLISTIC RESEARCH LABORATORY
ABERDEEN PROVING GROUND, MARYLAND

NOTICES

Destroy this report when it is no longer needed. DO NOT return it to the originator.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161.

The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The use of trade names or manufacturers' names in this report does not constitute indorsement of any commercial product.

UNCLASSIFIED**REPORT DOCUMENTATION PAGE**Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE September 1990	3. REPORT TYPE AND DATES COVERED Final, 1 Feb - 1 Jun 90	
4. TITLE AND SUBTITLE Small Force Effectiveness of Direct Fire Weapons: Model Analysis			5. FUNDING NUMBERS C: DAAL03-86-D-001	
6. AUTHOR(S) Harry L. Reed, Jr.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Battelle-Columbus 505 King Avenue Columbus, Ohio 43201			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) US Army Ballistic Research Laboratory ATTN: SLCBR-DD-T Aberdeen Proving Ground, MD 21005-5066			10. SPONSORING/MONITORING AGENCY REPORT NUMBER BRL-CR-641	
11. SUPPLEMENTARY NOTES The Contracting Officer's Technical Representative (COTR) for this report was Dr. Philip M. Howe of the Anti-Armor Munitions Technology Office, APG, MD. Task was performed under a Scientific Services Agreement issued by Battelle, Research Triangle Park Office, 200 Park Drive, P.O. Box 12297, Research Triangle Park, NC 27709.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This report discusses three modified versions of the TANKWARS model: 1) a version which uses an input table of priorities that allows the option of breaking off firing for newly appearing high priority targets and allows prioritization among targets that are already engaged by the various members of the team and that are of various significance; 2) a version which allows a more realistic meeting engagement in which both sides can advance and also have defending vehicles remain in hull defilade; and 3) a version in which the defending forces can operate in a "pop up" mode. In this mode, a nonfiring vehicle detects targets and transfers them to fully defilade tanks which then pop up to hull defilade, fire, and pop down. <i>Keywords</i>				
14. SUBJECT TERMS TANKWARS, System Analysis, Tanks, Tank Guns. (S) (S) (S)			15. NUMBER OF PAGES 93	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT SAR	

UNCLASSIFIED

INTENTIONALLY LEFT BLANK.

TABLE OF CONTENTS

	<u>Page</u>
1. INTRODUCTION	1
2. OBJECTIVE	1
3. TANKWARS	2
4. MODIFICATIONS TO TANKWARS	4
4.1 The Meeting Model	5
4.2 Priority Scheme	6
4.3 Pop-Up Defenders	9
4.4 The Vulnerability Model	11
5. ROUGH SPOTS IN THE PROGRAM TWMEET	16
6. CONCLUSIONS	17
7. REFERENCES	19
APPENDIX A: THE PROGRAM TWMEET	21
APPENDIX B: CODE FOR POP-UP MODIFICATIONS	69
APPENDIX C: FORTRAN CODE FOR PREPARATION OF VULNERABILITY DATA	77
APPENDIX D: MISCELLANEOUS CODE FOR RUNNING	87
DISTRIBUTION LIST	93



Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

INTENTIONALLY LEFT BLANK.

1. INTRODUCTION

This report discusses work supported by the Scientific Services Program through Battelle and the U.S. Army Research Office. The Contracting Officer's Technical Representative (COTR) was Dr. Philip M. Howe, the Anti-armor Munitions Technology Office, AMC.

The objective of the effort was to generate an improved understanding of specific performance characteristics, improvements which offer high leverage in force effectiveness.

The tasks to be performed were:

(a) Modify codes to provide realistic treatment of accuracy, target acquisition, and survivability attributes of target and shooter tanks.

(b) Conduct sensitivity analyses to determine leverage each attribute described above exerts upon kill ratio.

The COTR is reporting the results derived from the voluminous data that were supplied under task (b) in a separate report and in the development of his recommendations for the Army's Technical Base Program in armaments. Since the data developed are of little use to anyone who is not aware of the classified issues involved, Dr. Howe decided that no useful purpose would be served in including those data in this report. Rather, he has decided that this report should discuss task (a) and be a useful guide for those who wish to use the programs as modified or the ideas associated with the modifications for any similar analyses.

2. OBJECTIVE

The basic methodology upon which this work is based is the TANKWARS model. This report will briefly discuss the relation of TANKWARS to the effort, issues which needed consideration, and changes that were made to TANKWARS to accommodate those issues.

However, as will be mentioned in section 4, the effort was more of a series of responses to questions than the development of a clean set of computer codes, which resulted in a series of patches to the existing code. The author has tried to present a relatively clean code in Appendix A, and hopes that this report supplemented by other documentation on TANKWARS (Bunn, to be published; Reed 1990) will be useful as a user's guide.

3. TANKWARS.

TANKWARS is a Monte Carlo computer simulation of engagements between two homogeneous mechanized forces. The model simulates individual weapon systems, and the engagements include search, detection, selection, firing, impact, functional destruction, disengagement, and reengagement. Nominally, it can handle up to 20 armored vehicles on each side. The computer program was written by Mr. Fred Bunn of the Ballistic Research Laboratory (BRL), Aberdeen Proving Ground, MD (Bunn, to be published).

The author had previously worked with TANKWARS and made some modifications thereto (Reed 1990). Some of those modifications have been included in this present effort. While the relevant major aspects of those changes will be discussed in this report, the reader may find the report on that work a useful adjunct.

The main code, as it finally evolved, is shown in Appendix A. There are two major modifications of the original TANKWARS code that relate to the removal of the consideration of "waves" and the basic organization of scenarios.

TANKWARS allows the consideration of sequences of engagements with the output of one engagement defining the available force for the next engagement, etc. This feature is primarily handled by two subroutines in TANKWARS called WAVES and NXWAVE. Since the work done in this effort did not involve the use of waves, the author found it useful to strip out that part of the code to provide a more simple framework for the work that needed to be done.

TANKWARS allows two basic scenarios:

- * A moving attack against a stationary defender.

* A "meeting engagement" between two fully exposed and stationary forces.

In the moving attack, the southern force advances northward toward the defending force which is placed in the north.

Two forces are defined as "red" and "blue" and three options are available:

- (a) Red attack against a blue defense.
- (b) Blue attack against a red defense.
- (c) A meeting engagement as defined above.

Throughout the running of TANKWARS, there is a continual reference to which of these three scenarios is being used to determine movement and posture of individual vehicles. For our work, we found some difficulties with this approach: a meeting engagement of two moving forces was impossible, and the mixing of stationary and moving vehicles on the southern side for an overwatch attack was awkward.

Therefore, we changed to an approach in which each vehicle is individually assigned a role (as a moving attacker, a hull defilade defender, etc.). The vehicles on the north and the south are distinguished by the model only by their characteristics as defined by input files supplied by the analyst; and by the rule that southern tanks move northward, if they move at all, and northern tanks move southward, if they move at all. As a matter of fact, the blue force is placed in the north, and the red force is placed in the south by the model.

Smoke is not played well in TANKWARS; so it was not used in our studies and has been stripped out of the code in Appendix A. Likewise, we did not consider missiles, and they too have been removed for clarity.

The terrain models usually used for the representation of intervisibility give the statistical distribution of segments for which a moving target is in, and is out of line sight of a firing position. In particular, we have been using the model based on terrain at Hunfeld, Germany. There do not seem to be data for intervisibility between two vehicles, neither of which is operating from a previously selected position. In the absence of such data, we modified the use of the Hunfeld data, as discussed in the next section, by assuming that each vehicle disappears with the same statistics, and that line-of-sight between them requires that neither is in a "disappearance segment."

Finally, TANKWARS provides two "hard-wired" priority schemes to decide which target is highest in priority for engagement and three possible disengagement criteria (shoot to kill, disengage after a hit, or disengage after either a fixed number of rounds or a kill). We felt the need for more flexibility, including the ability to disengage a low priority target in favor of a target of higher priority; therefore, we added a more flexible approach for handling priorities and disengagements. The results of changing the priority scheme led to a parametric explosion and to not very exciting results. This seems to be more of a issue to be looked at after other issues are settled, rather than something to be looked at before more hardware-oriented issues are settled.

4. MODIFICATIONS TO TANKWARS.

Appendix A contains the code for the program called TWMEET, which is based on TANKWARS and allows moving meeting engagements among other scenarios. This program contains all the changes that were made to TANKWARS for use in this effort, except the changes for allowing the defending tanks to pop up from full defilade to hull defilade. The main control of the program is contained in the subroutine FORCES, which now does all of the control that was spread among FORCES, WAVES, and NXWAVE in TANKWARS.

Also, the code for smoke and missiles has been removed as has that associated with the "creation" of bullets. All aspects of firing are handled by the subroutines FIRE, FRDSSG, IMPACT, MAYHIT, and KILL.

4.1 The Meeting Model. As mentioned above, we gave up the meeting, red attack, and blue attack scenario approach in TANKWARS and substituted a more symmetric approach.

A new global variable called role(i) has been created. It defines the basic exposure and movement characteristics for each vehicle. For example, a defender cannot move, and remains in hull defilade unless he is killed, in which case he is allowed to hide if he is not killed; an attacker can move (in the direction toward the enemy), stop to shoot if appropriate, and is either fully exposed or in full defilade as he passes through the terrain. Another global variable sense(i) defines whether a vehicle is basically traveling north (+1) or south (-1).

The first three lines of the input game file (see Appendix D), which defined the number of tanks involved in the original three types of scenarios, are now replaced by two lines which define the number of blue attackers and defenders on line one, and the number of red attackers and defenders on line two.

The defenders could either be thought of as the original defenders or as tanks in overwatch which cover the advance of the attackers on the same side.

Terrain intervisibility is handled by allowing each side to have its own set of intervisibility intervals (the vector variable d[40] is replaced by the matrix variable d[2,40]). Any vehicles that move on a side are subject to disappear and reappear as defined by the distribution of intervals for that side. Two vehicles from different sides can see each other only if they are both in an "appear state."

The major changes associated with this change are contained in the subroutines:

- DEPLOY. Defenders are given zero velocity and attackers are given their initial velocity.
- INIT and INIT2. The roles of the tanks are used to define exposure (HD or FE) and motion (STATNY or MAXVL).

- INPUT. The first two lines of the game file are used to define the values for role(i), the sense(i), and the numbers of blue attackers (nbatt) and defenders (nbdef), and red attackers (nratt) and defenders (nrdef).
- PATH. The sense variable has been added to make the kinematics correct for tanks moving southward.
- SERCH1. The routine has been changed to make the decision as to when the first detection might occur, be independent of the concept of north and south. This is specially important since the original code wasn't required to handle an attack from the north.
- TERAIN. Values for terrain increments are defined for each side rather than for one side (d[2,40] replaces d[40]).

Minor changes are in subroutines:

- ACCELF and SLOWUP. The absolute value of velocity is used to figure the time required to complete the change in motion (dt).
- BLKDATA. The data definition for MEETING, RATTAK, and BATTAK were eliminated. That for DEFEND, ATTACK, etc. were added. (Note that OVERW and DECOY are not presently implemented. Of course, overwatch is presently handled by using DEFEND.)
- CANGO. The value assignment for the variable isatkr has been redefined.
- APPEAR, VANISH, and VANTER. The variable d(2,40) has been added.

4.2 Priority Scheme. We were concerned that, as we allowed tanks to exchange data on targets and as the ability of tanks to kill targets improved, we might tie up too many tanks on the same targets and thus, not properly handle newer high priority targets as they appear. To allow the tanks to break off engaging lower priority targets in favor of newer and higher priority tanks, we introduced a new priority scheme.

The following subroutines were changed to accommodate the new priority scheme:

•FRDSSG. In this routine, the decision to fire another round at the target is under the control of the priority system. The priority table (defined by the analyst) reflects the kinds of priority issues previously considered by TANKWARS (is the target firing, is it close, etc.), whether the firer is already firing at the target, and if other tanks on his side are firing at the target. Note that the original scheme is restored if overriding priority is given to target already engaged by the firer and if no priority difference is given for whether or not other tanks are firing at the target. Note that this also allows a more elaborate implementation of the concept handled by the variable share in the original TANKWARS - this allowed the option of prohibiting firing at the same target.

•PRIORN. The change here is to remove the consideration of the variable share. As mentioned in the discussion of FRDSSG, this function is handled by the priority table at least to the extent that firing at the same target can be given very low priority.

•PRIORT. The change here is to use the priority table (one is supplied for each side by the analyst) to define the priority of each target presented to PRIORT. The table as implemented at this stage of development has 192 entries.

The arguments for the table are:

- n6 has 4 values

1 = I am firing at the target already, and no one else is.

3 = I am firing at the target, and so is someone else.

2 = I am not firing at the target, nor is anyone else.

4 = No one is firing at the target.

- n5 has 3 values

3 = This is a new target for me.

2 = This is an old target which I have already hit.

1 = This is an old target which I have missed before.

- n4 has 2 values

1 = The target is within recognition range.

2 = The target is beyond recognition range.

- n3 has 2 values

1 = The target fired recently.

2 = The target didn't fire recently.

- n2 has 2 values

1 = The target has a target.

2 = The target doesn't have a target.

- n1 has 2 values

1 = The target is slowing or stationary.

2 = The target is moving.

The priority is given by

$$L = \text{lpri}(n1, n2, n3, n4, n5, n6, j),$$

where j defines the side of the firer, and $lpri$ is the level of priority table. Note that some of the entries in the table are wasted since $n2$ has the same priorities if $n1 = 2$ (if the target is beyond recognition range, it is impossible to tell if it has a target).

- RDMISC. The priority tables for the two sides are read in a change toward the end of the subroutine.

A typical priority table is shown in Appendix D.

4.3 Pop-up Defenders. Appendix B contains the code changes to the conventional version of TANKWARS that are necessary to produce the pop-up model. Some straightforward, but tedious work could add this feature to TWMEET; the command and control vehicle (blue No. 1 in the pop-up model) could be added as a new role, as could the actual pop-up defenders.

In this model, the blue side is in defense, and all the blue defender vehicles are in the pop-up mode, except blue No. 1, which is a command and control (c2) vehicle (entity). The pop-up defenders do not acquire targets. The c2 vehicle does not shoot, is never detected, does not die, and detects targets for the other defenders. Initially, all pop-up vehicles are in full defilade. As the c2 vehicle sees targets, it assigns them to the other defenders who pop up to hull defilade with a time constant $tpop$, fire up to $npop$ rounds (less if the target is killed soon enough), pop down again with a time constant $tpop$, and take a time $tmove$ (in which they would nominally move to another firing position) before becoming available for another assignment from the c2 vehicle. In the code, the vehicle that has popped up is presented with all the targets available to the c2 vehicle at that time and engages the most threatening. Note that the variable $busy(i)$ is used to keep the tank from selecting targets while it is exposed when popping up and down. A new variable, $ready(i)$, is used to keep it from being assigned targets while it is in full defilade and has not yet completed the moving process (time $tmove$ has not passed since popping down).

Routines that have been changed are:

- MAIN. In this routine, a file called `pop.dat` is open and read. This file contains the values of $npop$, $tpop$, and $tmove$. (See Appendix D for the files necessary to run the pop-up model.)

- DEATHS. In the loop "DO 20," i is started at 2 rather than 1, since blue No. 1 cannot die.
- DETECT. If blue No. 1 detects a target, the subroutine ASSIGN is called.
- DISENG. This routine has been changed to allow any pop-up vehicles to be "turned-off" so that they will not accept targets until after they have cycled through the pop-down process. (The busy(firer) = .true. is specially important in this regard.)
- EVENTS. This subroutine has been changed to allow the subroutines POPDWN, POPUP, and STANBY to be called as scheduled on the event queue.
- FINISH. This subroutine updates statistics at end of a single engagement. It has been changed to keep the c2 vehicle out of the exchange ratio and the win/loss decision.
- FRD SSG. This subroutine schedules effects after firing single shot gun. It has been changed to discontinue firing and schedule popping down if the tank has fired npop rounds.
- INIT. This subroutine begins the process of initializing the tanks on each side at the start of each battle. It has been changed to set the pop-up tanks to the ready-for-assignment status.
- INIT2. This subroutine is part of the initialization process. It has been changed to put all the pop-up tanks initially in full defilade and, of course, to initialize them as stationary.
- SEARCH. This subroutine has been changed to keep the southern tanks from seeing the c2 tank (i.e., blue No. 1).
- ASSIGN. This is a new subroutine. The c2 vehicle has at least one target available for servicing and has activated this tank(i). The tank is no longer ready for other assignments (to prevent purpose tremors), it is set to busy, it appears in hull defilade, and it is scheduled to have completed the pop-up process at $t + t_{pop}$. Note that the tank is assumed to be in hull defilade during the entire time interval t_{pop} .

•POPUP. This is a new subroutine. The tank is presumed to have completed popping up after an assignment (the time tpop has passed after assignment). The targets that the c2 vehicle has in its detection queue are transferred to the tank, it is set to be not busy, and it starts the process of selecting a target.

•POPDWN. This is a new subroutine that is activated after the time tpop has passed, and the tank is presumed to have achieved cover. Note that the tank is assumed to be in hull defilade during the entire time interval tpop. The tank goes into full defilade, and the subroutine STANBY is scheduled for tmove later, at which time, the tank is presumed to have moved to a new firing position.

•STANBY. This is a new subroutine that allows the tank to accept assignments now that the time tmove has passed after popping down.

4.4 The Vulnerability Model. This is essentially the same model as reported on in Reed (1990). It has been changed to allow both the target and firing vehicle to be in three states of motion: stationary, moving, and maneuvering.

The preprocessor has also been changed (see Appendix C) to allow accuracy to vary with range (this has no impact on the main code). The FORTRAN program for the preprocessor is given in the appendix. The basic preprocessor running under an interpreter on a PC became too slow; so we replaced it with a FORTRAN program which compiles under FORTRAN77 in a UNIX operating system.

However, the process of selecting maneuver or just moving for the tanks is still awkward. In the basic code for TWMEET, the target is not allowed to maneuver. In particular, the variable manuvr(i) is set to false in the subroutine INIT. To run cases where the vehicles maneuvered, we had to change assignment to

manuvr(i) = .true.

in the source and recompile.

We also consider the option in which each vehicle did not maneuver until it detected a target - at which time it went into the maneuver mode for the remainder of the engagement. This was accomplished by adding the assignment

`manuvr(i) = .true.`

in the subroutines DETECT and PINPNT and compiling a new version of the program.

Since the vulnerability model is an important change in TWMEET, the following discussion of that model is included much as it appears in Reed (1990).

The original TANKWARS handled the consideration of the azimuthal orientation of hits on the vehicles by creating a distribution of attack geometries. The advancing force proceeded on a course that for each Monte Carlo replication had a orientation that was randomly chosen (usually cardioid) about the axis that joined the center of the target array and the center of the attacking force. Thus, the tanks often marched in a direction that resulted in the forces never coming very close. This led to a number of indecisive replications within a sequence of Monte Carlo runs.

AMSAA does not have this feature in GROUNDWARS. The attackers advance toward the target, and they apply a random angle at the time of each impact.

TANKWARS and GROUNDWARS use two tables - a table of dispersions for different cases and ranges, and a table of conditional kill probabilities for different kill criteria, dispersions, ranges, exposures, and angles of attack. The probability of hit is calculated using normal distributions for hits on the turret and chassis, and then the conditional kill probabilities are multiplied by the hit probability to obtain the probabilities of kill for that instance. These kill probabilities are compared against a random number and one (or none) is selected.

We have adopted a scheme that has the attackers advancing directly toward the defenders as does GROUNDWARS; but we also do a lot of preprocessing of the data to account for the angular distribution of hits. The rationale follows:

The conditional kill tables are large, having some 6000 entries.

The hit probability/probability calculations involve Gaussian function calculations.

We had hopes that we eventually could add more than one type of ammo for each side, so that the memory requirements would get large.

Almost all the shooting was either by stationary vehicles shooting at moving vehicles or vice versa. The issue of the difference between the probability of hit for first and for subsequent rounds in stationary fire against stationary targets was moot (this involves random bias and random dispersion).

The approach was to create a table of hit and kill probabilities (given a shot) as functions of range for:

- * the nine (note that this is a change from Reed [1990]) cases of: 1) stationary shooting at stationary targets, 2) stationary shooting at moving targets, 3) stationary shooting at maneuvering targets, 4) moving shooting at stationary targets, 5) moving shooting at moving targets, 6) moving shooting at maneuvering targets, 7) maneuvering shooting at stationary targets, 8) maneuvering shooting at moving targets, and 9) maneuvering shooting at maneuvering targets;
- * two target conditions of:
 - hull defilade
 - fully exposed;
- * five levels of kill:
 - hit
 - mobility kill
 - fire power kill
 - mobility and firepower kill
 - catastrophic kill.

This, along with nine range values (0 to 4,000 m by 500 m), gives a table with 810 entries and puts a lot of mathematical calculation outside the Monte Carlo replications. The only drawback seems to be the need to do something about the correlation of impacts for the stationary/stationary case, should that become important.

The changes to accommodate this kill model are in the subroutines DAMAGE, INPUT, KILL, MAYHIT, and RDPKH. Note that the subroutines ACCERR, ACCMS, ACCSM, ACCSS, IZHIT, and RDEROR are not needed.

Most of the changes are straightforward and relate to shortening the calculation by avoiding specific efforts to calculate hit probability.

One piece of the code is worth discussing, particularly to provide an understanding of the preprocessing code in Appendix C.

It is in the subroutine KILL

```
temp = ranu(0.0)  ranu returns a random number
IF (temp .gt. p(1)) THEN
c no hit and no kill
    hit = .false.
    injury = ALIVE
ELSEIF (temp .gt. p(2)) THEN
c a hit and a "k" kill
    hit = .true.
    injury = KKILL
ELSEIF (temp .gt. p(3)) THEN
c a hit and an "m&f" kill but no "k"
    hit = .true.
    injury = MFKILL
ELSEIF (temp .gt. p(4)) THEN
```

c a hit and an "f" kill but no "m" kill

hit = .true.

injury = FKILL

ELSEIF (temp .gt. p(5)) THEN

c a hit and an "m" kill but no "f" kill

hit = .true.

injury = MKILL

ELSE

c a hit but no kill

hit = .true.

injury = ALIVE

ENDIF

In this section of code, the random number (temp) is located within a collection of segments in the unit interval which represents various mutually independent outcomes of the hit or miss.

p(1) is the probability of a hit; so if $1.0 > \text{temp} > p(1)$, the round missed.

p(2) is the probability of any kill less than a K (catastrophic) kill; so if $p(1) > \text{temp} > p(2)$, the round hit and achieved a K kill.

p(3) is the probability of any kill that is not both a mobility and a firepower kill; so if $p(2) > \text{temp} > p(3)$, the round hit and achieved both a mobility and a firepower kill.

p(4) is the probability of a mobility kill, but not a firepower kill; so if $p(3) > \text{temp} > p(4)$, the round hit and achieved a firepower kill but not a mobility kill.

p(5) is the probability that the hit produced no kill; so if $p(4) > \text{temp} > p(5)$, the round hit but did not kill.

With some thought the reader should be able to convince himself that:

$$1.0 \geq p(1) \geq p(2) \geq p(3) \geq p(4) \geq p(5) \geq 0.0$$

5. ROUGH SPOTS IN THE PROGRAM TWMEET

As mentioned above, while TWMEET is a fairly complete code, it still has some rough spots. Of course, this will always be the case in a situation such as ours in which we were continually asking new questions not covered by the code as it existed at the times the questions were asked.

One major rough spot is the fact that the pop-up model is not included in TWMEET. It seems reasonable that the idea of roles in TWMEET could be combined with some of the pop-up subroutines to obtain a good representation of popping up in TWMEET.

The handling of maneuvering targets by changes in the program and subsequent recompilation is not very satisfactory for any further parametric studies. Whether the vehicles maneuver or not should be handled by input through the miscellaneous files.

Decoys can be put back in TWMEET by using roles and the expedient of TANKWARS in which decoys are vehicles that either don't shoot or shoot, but don't produce any impacts.

We didn't use missiles, and therefore, they are not shown in the code. However, they could be put back essentially as they are represented in TANKWARS.

Smoke is not handled well in TANKWARS. There is a smoke model in GROUNDWARS, which AMSAA considers to be valid. However, GROUNDWARS has an approach to target detection which is very different than that employed in TANKWARS, and it is at least the opinion of this author that the role concept of TWMEET could much more easily be added to GROUNDWARS than the detection (and smoke) model of GROUNDWARS could be added to TANKWARS. (See Schmidt et al. 1989 for a discussion of GROUNDWARS).

Finally, the output is not very clean. It was acceptable for us since we were familiar with the program. However, it should be cleaned up before a more casual user employs it. For one thing, the blue force is always called the defender, and the red force is always called the attacker - even in meeting engagements. While these are only labels, they could be confusing. Also at present, the only results that are printed are the number of tanks killed on each side, the exchange ratio, the percent wins for each side, and the average number of rounds fired per vehicle by each side.

6. CONCLUSIONS

TANKWARS and the other variants we have employed seem to be very useful tools for looking at the tactical implication of engineering changes in tanks and similar combat vehicles. The code is reasonably amenable to changes and gave about the right level of detail and tactical context for the studies of a number of variations in armament systems for future tank concepts. In particular, we were able to look at the acquisition process, the fire control, the delivery accuracy, and the lethality of these systems in the context of small unit operations.

TWMEET and the concept of roles seems to offer a useful flexibility in the representation of forces and can be easily incorporated in TANKWARS and in GROUNDWARS (which is a child of TANKWARS).

INTENTIONALLY LEFT BLANK.

7. REFERENCES

Bunn, Fred. Unpublished paper on TANKWARS. U.S. Army Ballistic Research Laboratory, Aberdeen Proving Ground, MD.

Reed Harry L., Jr. "TANKWARS for the Parametric Consideration of System Concepts." Battelle Report No. DAAL03-86-D-0001, 28 February 1990.

Schmidt, Michael C., Gary R. Comstock, Lilly D. Harrington, and Barry J. Burns. "GROUNDWARS 4.0 User's Guide." AMSAA Technical Report, October 1989.

INTENTIONALLY LEFT BLANK.

**APPENDIX A:
THE PROGRAM TWMEET**

INTENTIONALLY LEFT BLANK.

Appendix A

The Program TWMEET

A.1 The File meeting.h

- c The file meeting.h is included by TWMEET. It defines the global
- c variables and is similar to common.h which is used with TANKWARS

implicit integer(i-n), real(a-h,o-z)

parameter (NN=20)

character*4 color

character*1 kview

integer ALL, NULL, FLS TGT

integer FD, HD, FE

integer BLU, RED

integer ALIVE, MKILL, FKILL, MFKILL, IKILL, KKILL

integer SLOWNG, STATNY, ACCELG, MAXVL

integer ATTACK, DEFEND, OVERW, DECOY

integer role, tactic, army

- c Change March 23, 1990 by HLReed for popup model

logical fot, kncels, ready

logical busy, empty, foes, los, seen, serchg, repeat

logical istest, share, xxfer

- c Change 5-11-90 to allow change in maneuver status

logical manuvr

real INFINT

c

common /aspekt/ angles(15), pangle(15), iangd

common /charc / color(2), kview(2)

common /consts/ PI, TWOPI, DEG, VNORTH(3)

common /const2/ ALL, NULL, FLS TGT,

1 FD, HD, FE, TURRET, HULL, BLU, RED, MEETNG, RATTAK,

2 BATTAK, ALIVE, MKILL, FKILL, MFKILL, IKILL, KKILL ,

3 SLOWNG,STATNY,ACCELG,MAXVL,INFINT,

4 ATTACK, DEFEND, OVERW, DECOY

common /contrl/ nreps,keyd(20),keym(20),scene,tmax, meth sm

common /cpath / nmaxt(2),accel(NN),decel(NN),ishtfs(2),

1 speed(NN), angle(2)

common /crandm/ irandm, jrandm

common /cshot/ kshot(2,20)

common /ctrace/ trace

logical trace

common /endgam/ sysdim(2,8), nang, ndisp

- c common /errors/ ssrgs(2,10), smrgs(2,10), velms(2,20),

- c 1 sserrs(2,16,10), smerrs(2,16,10), addons(2,2,20), nadds(2)

```

common /error2/ rex, rey, reliab(2)
c Change 12-30-89 by HL Reed to allow HD pop down and up.
common /fcycle/ nrds(2),nrpt(2),nipods(2),nrpb(2),tactic(2),
1   tof(2,8),trelod(2), tfirst(2,8), tmedin(2), tfixed(2,8),
2   rof(2),kind rd(2),tbump(2),nbump(2),thide(2),tmin(2),
3   nprior(2), share(2), xxfer(2),
4   npop, tpop, tmove
common /n sys/ ntanks(4,6),nblu,nred,nbatt,nbdef,nratt,nrdef
common /sensor/ psense(2,8), pinfin(2,3,10), tbar(2,3,10),
1   ndets(2), tlook(2), pinp(2), repeat, recknz(2), pfalse(2,2),
2   pntime(2)
common /states/ army(NN), role(NN), sense(NN),
1   busy(NN), empty(NN), fot(NN,NN), foes(NN,NN), ichg(NN),
2   knceal(NN), knceals(NN,NN), know(NN,NN),
2   life(NN), los(NN,NN), mot(NN,NN), motion(NN), mslfly(2,NN,5),
3   nhot(NN), nbrst(NN), ndet(NN), nrd(NN), nrrib(NN), nipod(NN),
4   nrot(NN), nrtgt(NN), rgvis(3,NN),
5   seen(NN,NN), serchg(NN), tfire(NN,NN), tfire2(NN), vbx(NN),
6   vby(NN), t0(NN), x0(NN), y0(NN), vx0(NN), vy0(NN),
7   xp(NN), tlast(NN), ready(NN), manuvr(NN)
common /state2/ idecoy(NN), iflash(NN), ndecoy(2), nflash(2)
common /stats/ statb(8), mystat(4)
common /vars6/ irginc, rgincr, rginc2
common /csmoke/ tsmoke(20),psmoke(20,3), invisb
common /smoke1/ touti1(21,5),toutv1(21,5),touti(21,5),
1   toutv(21,5),tini(21,5),tinv(21,5),ptbl(21),rtbl(5)
common /v16a/ krep
common /v17a/ istest
common /v23/ nwave, nwaves, nsurv, neval, nused(3000), nreps3,
1   statc(8), noammo, loammo, noamo2, loamo2
common /v24/ nstats(5,2)
common /where/ min rg, max rg, inc rg
common /where2/ nrg, rg0, rg, s(3), vt(3), vf(3)
common /fitnes/ quit(2,2), alloc(2,6), fit(NN,6)
c   change by HLReed to add new priority scheme 2-11-90
common /prty/ lpri(2,2,2,2,3,4,2)

```

A.2 The File clock.h

```

c   The file clock.h is also included by TWMEET. It provides the
c   storage for the event queue.
parameter (NE=2000)
character*6 what
integer who, whom
logical prflag
common /event1/ what(NE)
common /event2/ when(NE), who(NE),

```

```

1  whom(NE), next(NE), nxevnt, nxidle, prflag
   save /event1/, /event2/

```

.bp

A.3 The Program File twmeet.f

```

c  MAIN ROUTINE
c  Main: read input and simulate scenarios.
   include 'meeting.h'
c
   call input
   call forces
   END
c
   SUBROUTINE ACCEL F (t, firer)
c  Accelf: simulate tank starting to accelerate.
   include 'meeting.h'
   integer firer
1  format (f8.2,1x,a4,i3,' speed up',9x,'(was slowing)')
2  format (f8.2,1x,a4,i3,' speed up',9x,'(was halted)')
3  format (f8.2,1x,a4,i3,' speed up',9x,'(was speeding up)')
4  format (f8.2,1x,a4,i3,' speed on',9x,'(is cruising)')
c
   if (keyd(4).gt.0) print *, '>accel'
   if (life(firer).ne.FKILL.and.visb.eq.1.and.knceal(firer).ne.FD)
1  call skedul (t,firer,'vanish',NULL)
   narmy = army(firer)
   IF (motion(firer).eq.SLOWNG) THEN
c  Previous motion was slowing
   if (keyd(1).ge.2) print 1, t, color(narmy), firer
   call path(firer,t,motion(firer),0.0,x,y,vx,vy)
   dt = (speed(firer)-abs(vy))/accel(firer)
   call skedul (t+dt,firer,'maxvel',NULL)
   motion(firer) = ACCEL G
   ELSE IF (motion(firer).eq.STATNY) THEN
c  Previous motion was stationary
   if (keyd(1).ge.2) print 2, t, color(narmy), firer
   call path(firer,t,motion(firer),0.0,x,y,vx,vy)
c  schedule time full velocity reached (max vel)
   dt = speed(firer)/accel(firer)
   call skedul(t+dt,firer,'maxvel',NULL)
   motion(firer) = ACCEL G
   ELSE IF (motion(firer).eq.ACCEL G) THEN
c  Previous motion was accelerating
   if (keyd(1).ge.2) print 3, t, color(narmy), firer
   ELSE IF (motion(firer).eq.MAXVL) THEN
c  Previous motion was cruising at max velocity
   if (keyd(1).ge.2) print 4, t, color(narmy), firer

```

```

ENDIF
if (keyd(4).gt.0) print *, '<accel'
END

c
FUNCTION ANGLEF (a, b)
c   Anglef: find angle between two vectors.
dimension a(3), b(3)
c
vabsa = sqrt( dot( a,a ) )
vabsb = sqrt( dot( b,b ) )
dotab = dot( a,b )
dm = dotab/(vabsa*vabsb)
dm = amin1(1.,amax1(-1.,dm))
dm = acos(dm)
r3 = a(1)*b(2) - a(2)*b(1)
anglef = -sign(dm,r3)
END

FUNCTION ANGSUM (a, b)
c   Angsum: add 2 angles and adjust answer to lie between +-PI.
c = a+b
10  IF (c.lt.-180.) THEN
    c = c+360.
    GOTO 10
ELSE IF (c.gt.180.) THEN
    c = c-360.
    GOTO 10
ENDIF
c   angle is adjusted.
angsum = c
END

c
SUBROUTINE APPEAR(t,tgt,firer)
c   Appear: if tgt appears treat, otherwise reschedule appearance
include 'meeting.h'
integer tgt,firer, armyf, armyt
common /terane/ d(2,40), xold(20), yold(20), dist(20), iseg(20)
rss(x,y) = sqrt(x*x+y*y)
1  format(f8.2,1x,a4,i3,' appears ',9x,'(x=',f8.1,' y=',f8.1,')')
2  format(f8.2,1x,a4,i3,' LOS to ',a4,i3,' starts.')
c
if (trace) print *, '>appear'
armyt = army(tgt)
armyf = 3-armyt
IF (invisb.eq.1) THEN
    if(speed(tgt).le.0.)print *, 'APPEAR: armyt,speed=',armyt,
1    speed(tgt)
    if(speed(tgt).le.0.) stop
    call path(tgt,t,motion(tgt),0.2,x,y,vx,vy)
c   Terrain causes intermittent LOS.

```

```

travel = rss(x-xold(tgt), y-yold(tgt))
IF (travel.gt.dist(tgt)) THEN
c   Tgt is no longer masked by terrain
    if (keyd(1).gt.1) print 1,t,color(armyt),tgt,x,y
    xold(tgt) = x
    yold(tgt) = y
    iseg(tgt) = iseg(tgt)+1
    if (iseg(tgt).gt.40) iseg(tgt)=iseg(tgt)-40
    dist(tgt) = d(armyt,iseg(tgt))
    call aprter(t,tgt,firer,FE)
c   Schedule next disappearance
    dt = dist(tgt)/speed(tgt) + 0.01
    call skedul(t+dt,tgt,'vanish',NULL)
ELSE
c   Still masked by terrain, so reschedule mask end
    IF (life(tgt).eq.ALIVE) THEN
        dt = (dist(tgt) - travel) / speed(tgt) + 0.01
        call skedul (t+dt,tgt,'appear',NULL)
    ENDIF
ENDIF
ELSE
    print *, 'smoke not played'
ENDIF
if (trace) print *, '<appear'
END

c
SUBROUTINE APRTER(t,tgt,firer,jexpos)
c   Apprter: Tgt has appeared from behind terrain, reset.
include 'meeting.h'
integer tgt,firer
common /terane/ d(2,40), xold(20), yold(20), dist(20), iseg(20)
1   format(f8.2,1x,a4,i3,' aprters ',9x,'(x=',f8.1,' y=',f8.1,')')
c
if (trace) print *, '>aprter'
narmy = army(tgt)
knceal(tgt) = jexpos
c   Restore all lines-of-sight involving tgt
DO 20 i=1,nblu+nred
    IF (knceal(i).ne.FD) THEN
        los(tgt,i) = army(i).ne.narmy
        los(i,tgt) = army(i).ne.narmy
    ENDIF
20  CONTINUE
c   Turn search on if it is off
    IF (.not.repeat) THEN
        repeat = .true.
        call skedul(t+.01,0,'search',NULL)
    ENDIF
if (trace) print *, '<aprter'
END

```

```

c      BLOCK DATA BLKDAT
      include 'meeting.h'
      data color,      pi,      twopi,      deg
1      /'Blue', 'Red ', 3.141592654, 6.283185308, 57.29577951/
      data VNORTH /0., 1., 0./
      data ALL, NULL, FLS TGT /0, 0, -1/
      data FD, HD, FE /1, 2, 3/
c      Added data for meeting model
      data ATTACK, DEFEND, OVERW, DECOY/1, 2, 3, 4/
      data BLU, RED /1, 2/
      data ALIVE, MKILL, FKILL, MFKILL, IKILL, KILL /1,2,3,4,5,6/
      data SLOWNG, STATNY, ACCELG, MAXVL, INFINT
1      / 1, 2, 3, 4, 1.e35/
      data keyd, keym /40*0/
      END

c      SUBROUTINE CANCEL (l, act, it)
c      Cancel: cancel 'act' events for 'l' entity.
c      (all events if act="")
c      Definitions of local variables:
c      m - pointer to previous event
c      n - pointer to current event being considered
      include 'clock.h'
      logical is what, is who, is whom
      character*6 act
1      format(9x,'cancel ',i3,' ',a6,i3,' at time',f8.2)
c
      m = 0
      n = nxevnt
10      IF (n.ne.0) THEN
c      Continue until n=0
      is who = l .eq.who(n)
      is what = act.eq.what(n) .or. act.eq.'all '
      is whom = it.eq.whom(n) .or. it.eq.0
      IF (is who .and. is what .and. is whom) THEN
c      Then remove event
      if (prflag )print 1, l, act, it, when(n)
      if (m.eq.0) nxevnt = next(n)
      if (m.ne.0) next(m) = next(n)
      next(n) = nxidle
      nxidle = n
      if (m.eq.0) n = nxevnt
      if (m.ne.0) n = next(m)
      ELSE
c      Don't remove event. Shift to next event.
      m = n
      n = next(n)
      ENDIF
      GOTO 10

```

```

ENDIF
END

c
LOGICAL FUNCTION CAN GO (firer, t)
c
Can go: True iff is stationary and can move.
include 'meeting.h'
integer firer
logical is atkr, m alive, faster

c
narmy = army(firer)
c
Change for meeting model
is atkr = role(firer).eq.ATTACK
m alive = life(firer).eq.ALIVE .or.
1 life(firer).eq.FKILL
faster = (motion(firer).eq.STATNY .or.
1 motion(firer).eq.SLOWNG)
can go = is atkr .and. m alive .and. faster
END

c
SUBROUTINE DAMAGE (t, l, it, injury)
C
Damage: schedule effects.
c
Changed May 18, 1989 for simplified hit and kill model, HL Reed
include 'meeting.h'
character*2 kt(6)
data kt /'no','M-','F-','MF','I-','K-'/
1 format(f8.2,1x,a4,i3,1x,'Hits ',a4,i3,' (no damage).')
2 format(f8.2,1x,a4,i3,1x,a2,'-kills ',a4,i3)
c
if (trace) print *, '>damage'
n=army(l)
m = 3-n
IF(keyd(1).ge.2) THEN
    if (injury.eq.1) print 1,t,color(n),l,color(m),it
    if (injury.gt.1) print 2,t,color(n),l,kt(injury),color(m),it
ENDIF

injold = life(it)
IF (injury.eq.KKILL .and. injury.nejold) THEN
c
Treat first catastrophic kill.
    life(it) = KKILL
    call damagf(t,it,m)
    call damagm(t,it)
    call cancel(it,'kill ',NULL)
    call newtgt (t,l,it)
    call deaths(t)
ELSEIF (injury.nejold .and. injold.lt.MFKILL) THEN
c
Treat new damage (less than catastrophic).
    IF (injury.eq.MKILL) THEN
        if (injold.eq.FKILL) life(it) = MFKILL
        if (injold.eq.ALIVE) life(it) = MKILL
    
```

```

        if (injold.eq.ALIVE .or. injold.eq.FKILL) call damagm (t,it)
    ELSE IF (injury.eq.FKILL) THEN
        if (injold.eq.ALIVE) life(it)=FKILL
        if (injold.eq.MKILL) life(it)=MFKILL
        call damagf(t,it,m)
    ELSE IF (injury.eq.MFKILL) THEN
        if (injold.lt.MFKILL) life(it) = MFKILL
        if (injold.ne.MKILL) call damagm (t, it)
        if (injold.ne.FKILL) call damagf (t,it,m)
    ENDIF
    if (life(it).eq.MFKILL.andjold.lt.MFKILL)
1      call skedul(t+tbump(n),it,'ikill ',NULL)
    ENDIF
    if (trace) print *, '<damage'
END

```

```

c
c  SUBROUTINE DAMAGF (t,it,m)
c  Damagf - Discard activities due to firepower kill.
include 'meeting.h'

```

```

        nrtgt (it) = 0
c      Cancel firings by target.
        DO 40 j=1,nblu+nred
            fot(it,j) = .false.
40      CONTINUE
        call cancel(it,'fire ',NULL)
        call cancel(it,'select',NULL)
        IF (life(it).eq.FKILL .and. speed(m).gt.0.0) THEN
            call cancel (it,'slowup',NULL)
            call cancel (it,'halt ',NULL)
            call cancel (it,'accel ',NULL)
            call skedul (t, it, 'accel ',NULL)
            dt = thide(m)
c      change for meeting model
            if(role(it) .eq. DEFEND) dt = 5.0
            call skedul (t+dt,it,'hide ',NULL)
        ENDIF
    END

```

```

c
c  SUBROUTINE DAMAGM (t, it)
c  Damagm - Simulate mobility kill on the tgt.
include 'meeting.h'
logical sos
c  sos - stopped or slowing
    if (trace) print *, '>damagm'
    call cancel (it, 'maxvel', NULL)
    call cancel (it, 'accel ', NULL)
    call cancel (it, 'hide ', NULL)
    sos = vabs(vt).le.0.0 .or. motion(it).eq.SLOWNG
    if (.not.sos) call skedul (t, it, 'slowup', NULL)

```

```

    if (trace) print *, '<damagm'
    END
c
    SUBROUTINE DEATHS (t)
c    Deaths: Find death toll on each side. A tank is considered
c    dead if it is I-killed, K-killed, or F-killed & hidden.
    include 'meeting.h'
    logical dead1, dead2
    integer dead(2)
1    format (i3, ' Blu dead, ', i3, ' Red dead.')
c
    if (trace) print *, '>deaths'
    dead(BLU) = 0
    dead(RED) = 0
    DO 20 i=1, (nblu-ndecoy(BLU))
        dead1 = life(i).ge.IKILL
        dead2 = knceal(i).eq.FD .and. life(i).ge.FKILL
        if (dead1 .or. dead2) dead(BLU)=dead(BLU)+1
20    CONTINUE
    DO 30 i=nblu+1, (nblu + nred - ndecoy(RED))
        dead1 = life(i).ge.IKILL
        dead2 = knceal(i).eq.FD .and. life(i).ge.FKILL
        if (dead1 .or. dead2) dead(RED)=dead(RED)+1
30    CONTINUE
    if (keyd(1).ge.2) print 1, dead
    if ((nblu-ndecoy(BLU)).eq.dead(BLU) .or.
1    ((nred-ndecoy(RED)).eq. dead(RED)))
1    call skedul(t+5., NULL, 'finish', NULL)
    if (trace) print *, '<deaths'
    END
c
    SUBROUTINE DEPLOY
c    Deploy: position & orient all tanks at beginning of engagement.
c    Changed for meeting engagement
    include 'meeting.h'

    if (trace) print *, '>deploy'
    spacng = 100.
    DO 20 n=1, nblu+nred
        t0(n) = 0.0
20    CONTINUE
c    position red (southern) tanks on the x-axis
    isouth = nblu + 1
    jsouth = nblu + nred
    x0(isouth) = -0.5*(nred-1)*spacng
    xp(isouth) = x0(isouth)
    y0(isouth) = 0.0
    IF(role(isouth).eq.ATTACK) THEN
        vy0(isouth) = speed(isouth)
    ELSE

```

```

        vy0(isouth) = 0.0
    ENDIF
    n = isouth
    if (keyd(1).ge.2) print *, n, x0(n), y0(n), 0.0, vy0(n)
    DO 21 n=isouth+1,jsouth
        x0(n) = x0(n-1) + spacng
        xp(n) = x0(n)
        y0(n) = 0.0
        IF(role(n).eq.ATTACK) THEN
            vy0(n) = speed(n)
        ELSE
            vy0(n) = 0.0
        ENDIF
        if (keyd(1).ge.2) print *, n, x0(n), y0(n), 0.0, vy0(n)
21    CONTINUE
c    position blue (northern) tanks on the x axis
    x0(1) = -0.5*(nblu-1)*spacng
    xp(1) = x0(1)
    y0(1) = rg0
    IF(role(1).eq.ATTACK) THEN
        vy0(1) = - speed(1)
    ELSE
        vy0(1) = 0.0
    ENDIF
    n = 1
    if (keyd(1).ge.2) print *, n, x0(n), y0(n), dm, vy0(n)
    DO 30 n=2,nblu
        x0(n) = x0(n-1) + spacng
        xp(n) = x0(n)
        y0(n) = rg0
        IF(role(n).eq.ATTACK) THEN
            vy0(n) = - speed(n)
        ELSE
            vy0(n) = 0.0
        ENDIF
        if (keyd(1).ge.2) print *, n, x0(n), y0(n), dm, vy0(n)
30    CONTINUE
    if (trace) print *, '<deploy'
    END

c
SUBROUTINE DETECT (t, firer, tgt)
c    Detect: find if tgt detected and schedule subsequent events.
include 'meeting.h'
integer firer, tgt, armyf, armyt
1    format (f8.2,1x,a4,i3,' detects ',1x,a4,i3)
c
    if (trace) print *, '>detect'
    armyf = army(firer)
    armyt = 3-armyf
    IF (los(firer,tgt) .and. .not.seen(firer,tgt) .and.

```

```

1  ndet(firer).lt.ndets(armyf)) THEN
    if(keyd(1).ge.2)print 1,t,color(armyf),firer,color(army),tgt
    ndet(firer) = ndet(firer)+1
    seen(firer,tgt) = .true.
    t human = 0.0*exp(rolln(0.5))
    call selecs(t,firer,thuman)
    IF(xxfer(armyf)) THEN
        i = 1
        if(firer .GT. nblu) i = nblu+1
        call skedul(t+2.,i,'xfer ',tgt)
    ENDIF
ENDIF
if (trace) print *, '<detect'
END

c
SUBROUTINE DET RG (narmy)
c  Det rg: Find the max ranges at which each firer in 'narmy' detects.
include 'meeting.h'
integer first, tank, cond
real p1, p2, r, r1, p
1  format (' Range to which tank can see',/,
1  ' Tank HD FE-S FE-M ranu')
2  format (i5,3f8.1,f8.4)
c
if (trace) print *, '>detrg'
if (keyd(1).ge.2) print 1
c  Find first and last firers on this side (narmy).
IF (narmy.eq.BLU) THEN
    first = 1
    last = nblu
ELSE
    first = nblu+1
    last = nblu+nred
ENDIF
c  Loop thru all tanks on the side
DO 80 tank = first,last
    p = ranu(0.0)
    DO 70 cond=1,3
        p1 = 1.0
c        Search for P-infinity values bounding x
        DO 60 krg=1,8
            p2 = pinfin(narmy,cond,krg)
            IF (p2 .lt. p) GOTO 65
            p1 = p2
60        CONTINUE
        p2 = 0.0
65        CONTINUE
c        Interpolate on p-infinity to find range.
        r1 = irginc*(krg-1)
        r = r1 + irginc*(p1-p)/(p1-p2)

```

```

        rgvis(cond,tank) = r
70    CONTINUE
        if (keyd(1).ge.2) print 2,tank,(rgvis(cond,tank),cond=1,3),p
80    CONTINUE
        if (trace) print *, '<detrg'
        END
c
        SUBROUTINE DIS ENG (t, firer, tgt,drop,take)
c        Diseng: attempt to disengage 1 firer from 1 target.
c        Diseng is called by impact if firer condition warrants.
c        When I include guns, other routines may call it.
        include 'meeting.h'
        integer armyf, armyt, tgt, firer
        logical in brst, hav amo, on tgt, drop, take, cango
3        format (f8.2,1x,a4,i3,' dis-engs ',a4,i3,20x,'#tgts=',i2)
c
        if (trace) print *, '>diseng'
c        Set useful local variables
        my tgt = nrtgt(firer)
        armyf = army(firer)
        armyt = 3-armyf
        hav amo = nrd(firer).lt.nrds(armyf)
        inbrst = nrpb(armyf).gt.1 .and. (0.ne.mod(nrpb(firer),
1        nrpb(armyf)))
        if (tgt.eq.FLS TGT) on tgt = .true.
        if (tgt.ne.FLS TGT) on tgt = fot(firer,tgt)
        IF (on tgt) THEN
c        Firer on this target
        kind = kindrd(armyf)
        IF (kind.le.2 .or. kind.eq.5) THEN
            IF (nrpb(armyf).le.1) THEN
c            Single shot gun system or STAFF fire & forget system.
                IF (tgt.ne.FLS TGT) THEN
                    if (fot(firer,tgt)) call cancel (firer,'fire ',tgt)
                    fot(firer,tgt) = .false.
                ENDIF
                hav amo = nrd(firer).lt.nrds(armyf)
                IF (hav amo) THEN
                    thuman = 0.*exp(rolln(0.5))
                    call selecs(t,firer,thuman)
                ELSEIF (can go(firer,t).and.ishtfs(armyf).gt.0) THEN
c                Firer moves on.
                    if(keyd(1).ge.2)print 3, t,color(armyf),firer,
1                    color(armyt),tgt,ndummy
                    call skedul(t,firer,'accel ',NULL)
                ENDIF
                nrot(firer) = 0
                nrtgt(firer) = 0
            ELSE
c            Burst fire gun system.

```

```

        print *, 'DISENG: Not implemented for burst fire guns.'
        STOP
    ENDIF
ENDIF
ENDIF
IF (.not.repeat) THEN
    repeat = .true.
    call skedul (t+.01,0,'search',NULL)
ENDIF
if (trace) print *, '<diseng'
END

c
FUNCTION DOT (a, b)
c   Dot: find dot product  a dot b.
c   dimension a(3), b(3)

c
    dot = a(1)*b(1)+a(2)*b(2)+a(3)*b(3)
END

c
SUBROUTINE ENGAGE (t1, t2, firer, tgt)
c   Engage: begin engagement of a new tgt by this firer.
    include 'meeting.h'
    integer armyf, armyt, firer, tgt
1   format(' ENGAGE: armyf,ishtfs,firer,motion,STATNY',8i3)
c
    if (trace) print *, '>engage'
    armyf = army(firer)
    armyt = 3-armyf
    IF (life(firer).lt.FKILL.AND.nrd(firer).lt.nrds(armyf) )THEN
        if(keym(18).gt.1)print 1,
1   armyf,ishtfs(armyf),firer,motion(firer),STATNY
        nbrst(firer) = 1
        IF (ishtfs(armyf).gt.0 .AND. motion(firer).ne.STATNY
1   .AND. speed(firer).gt.0.0) THEN
c           halt to fire
            call cancel (firer,'maxvel',NULL)
            call cancel (firer,'accel ',NULL)
            call skedul(t1,firer,'slowup',NULL)
        ELSE
c           Schedule a fire event otherwise
c           find range to target
            IF (tgt.eq.-1) THEN
                rg = rg0
                nrg = int(0.5+rg/irginc)
            ELSE
                dm = rgf(t1,tgt,firer)
            ENDIF
            nrg = min0(8,nrg)
            dt = tfirst(army(firer),nrg) * exp(rolln(0.5))
            nrib(firer) = 0

```

```

      nrot(firer) = 0
change 23 Nov 89 by HLReed to make sure a round has been loaded
      t3 = amax1(tfire2(firer)+tmin(armyf),t2+dt)
      call skedul(t3,firer,'fire ',tgt)
      ENDIF
    ENDIF
    IF (trace) print *, '<engage'
  END

```

```

c
SUBROUTINE EVENTS
c  Events: call each event in sequence.
  include 'meeting.h'
  character*6 iwhat
1  format (' EVENTS: No such event type. Event=',a6,' ',
1  ' Who=',i2,' Whom=',i2,' Time=',f7.2)

```

```

c
  if (trace) print *, '>events'
c  Initialize for battle
    call reset(keyd(5).gt.0)
    call init
    tm lst = 0.0
c  Perform all events in the battle
10  CONTINUE
    call nextev(iwho, iwhat, iwhom, t)
    IF (iwhat.eq.'search') THEN
      call search (t)
    ELSEIF (iwhat.eq.'vanish') THEN
      call vanish (t,iwho,iwhom)
    ELSEIF (iwhat.eq.'appear') THEN
      call appear (t,iwho,iwhom)
    ELSEIF (iwhat.eq.'detect') THEN
      call detect (t,iwho,iwhom)
    ELSEIF (iwhat.eq.'select') THEN
      call select (t,iwho)
    ELSEIF (iwhat.eq.'xfer ') THEN
      call xfer(t,iwho,iwhom)
    ELSEIF (iwhat.eq.'fire ') THEN
      call fire  (t,iwho,iwhom)
c  changed for meeting program
    ELSEIF (iwhat.eq.'impact') THEN
      call impact (t,iwho,iwhom)
    ELSEIF (iwhat.eq.'slowup') THEN
      call slowup (t,iwho)
    ELSEIF (iwhat.eq.'halt ') THEN
      call halt  (t,iwho)
    ELSEIF (iwhat.eq.'accel ') THEN
      call accel (t,iwho)
    ELSEIF (iwhat.eq.'maxvel') THEN
      call maxvel (t,iwho)
    ELSEIF (iwhat.eq.'ikill ') THEN

```

```

        call latekl (t,iwho,iwhom)
    ELSEIF (iwhat.eq.'hide ') THEN
        call hide (t,iwho)
    ELSEIF (iwhat.eq.'finish') THEN
        call finish (tm lst)
        GOTO 99
    ELSE
        print 1,iwhat, iwho, iwhom, t
        STOP
    ENDIF
    tm lst=t
    GOTO 10
99  if (trace) print *, '<events'
    END
c
    SUBROUTINE FINISH (t)
c    Finish: update statistics at end of a single engagement.
    include 'meeting.h'
    integer balive, ralive, brds, rrds
    dimension stata(8)
1    format(i6,2(5i3),4i3,1x,2f5.1,i9)
2    format(' Rep Status of Combatants ',
1 ' Rds Used Used/Tank',/
1 ' |---Blue---| |---Red---| ',
1 'by System Blue Red seed',/
1 6x,2(1x,'AL MO FO MF K'),2x,'1 2 3 4')
c
    if (trace) print *, '>finish'
c    Count surviving blues and rounds fired
    balive = 0
    brds = 0
    DO 10 i=1,nblu
        k = life(i)
        if (k.ge.5) k=k-1
        nstats(k,BLU) = nstats(k,BLU)+1
        if (life(i).lt.FKILL) balive = balive+1
        brds = brds+nrd(i)
10    CONTINUE
c    Count surviving reds and red rounds fired.
    ralive = 0
    rrds = 0
    DO 20 i=1,nred
        j = i+nblu
        k = life(i+nblu)
        if (k.ge.5) k=k-1
        nstats(k,RED) = nstats(k,RED)+1
        if (life(j).lt.FKILL) ralive = ralive+1
        rrds = rrds+nrd(j)
20    CONTINUE
    mystat(1) = mystat(1) + nblu-balive

```

```

        mystat(3) = mystat(3) + nred-ralive
c
DO 30 i=1,4
30   stata(i) = 0.0
      if (balive.gt.0 .and. ralive.eq.0) stata(1)=1.
      if (balive.eq.0 .and. ralive.gt.0) stata(2)=1.
      if (balive.gt.0 .and. ralive.gt.0) stata(3)=1.
      if (balive.eq.0 .and. ralive.eq.0) stata(4)=1.
      stata(5) = nblu-balive
      stata(6) = nred-ralive
      stata(7) = float(brds)/float(nblu)
      stata(8) = float(rrds)/float(nred)
      excha = 0.0
      if (stata(5).gt.0.0) excha = stata(6)/stata(5)
      DO 40 i=1,8
        statb(i) = statb(i)+stata(i)
40   CONTINUE
c
      if (keyd(1).ge.2 .or.
1     (krep.eq.1 .and. keyd(1).eq.1)) print 2
      if (keyd(1).gt.0) print 1, krep, nstats, (nrd(i),i=1,4),
1     stata(7), stata(8), irandm
      if (trace) print *, '<finish'
      END
c
SUBROUTINE FIRE (t,firer,tgt)
c   Fire: Simulate firing of a round & schedule effects.
      include 'meeting.h'
      integer armyf, armyt, firer, tgt
1     format(f8.2, 1x, a4, i3, ' fires at ', a4, i3)
2     format(f8.2, 1x, a4, i3, ' ran out of ammo. ')
c
      if (trace) print *, '>fire'
      busy(firer)=.false.
      IF (life(firer).ge.FKILL) THEN
        print *, 'FIRE: firer',firer,' is F-killed or worse.'
        STOP
      ELSEIF (tgt.eq.0) THEN
        print *, 'FIRE: firer',firer,' has no target.'
        STOP
      ELSE
c        Find nrs for tgt, army of firer, army of tgt
        armyf = army(firer)
        armyt = 3-armyf
        if (keyd(1).ge.2) print 1,t,color(armyf),firer,
1        color(armyt),tgt
c        Update last firing time for firer & for firer at this tgt
        if (tgt.gt.0) tfire(firer,tgt) = t
        tfire2(firer) = t
c        Update positions, velocities

```

```

      IF (tgt.eq.-1) THEN
        rg = rg0
        nrg = max0(1,int(0.5+rg/irginc))
        s(1) = 0.0
        s(2) = rg0
        s(3) = 0.0
        if (armyf .eq. RED) s(2) = -rg0
      ELSE
        dm = rgf(t,tgt,firer)
      ENDIF
c    Schedule any pinpoint detections
      call pinpnt (t,firer)
      IF (iflash(firer).eq.0) THEN
c    Branch for real firer (do nothing if firer is flashing decoy)
        tfly = tof(armyf,nrg)
        t2 = t+tfly
        kshot(armyf,1) = kshot(armyf,1) + 1
c    Schedule impact for bullet
        call skedul (t+tfly,firer,'impact',tgt)
c    Update stowed rounds and expenditure
        nrd(firer) = nrd(firer)+1
        nrrib(firer) = nrrib(firer)+1
        if(nrrib(firer).gt.nrpb(armyf)) nrrib(firer)=1
        nrot(firer) = nrot(firer)+1
c    Move, fire, or switch targets as required
        IF (kind rd(armyf).eq.1) THEN
          if (nrpb(armyf) .le.1) call frd ssg(t,firer,tgt,armyf)
        ELSE
          print *,'FIRE: kind rd',kind rd(firer),' unknown.'
        ENDIF
      ENDIF
    ENDIF
    if (keyd(1).ge.2 .and. nrd(firer).ge.nrds(armyf)) print 2,
1    t,color(armyf),firer
    if (trace) print *,'<fire'
    END
c
c    SUBROUTINE FORCES
c    changed for meeting to simplify code
c    Forces: loop through desired ranges.
    include 'meeting.h'
    dimension istatb(8)
    integer range0
    DO 30 range0 = min rg, max rg, inc rg
      rg0 = range0
      DO 5 i=1,20
        kshot(1,i) = 0
        kshot(2,i) = 0
5      CONTINUE
      nrg = rg0/irginc

```

```

DO 20 i=1,8
  statb(i) = 0.0
20  CONTINUE
DO 100 i = 1,nreps
  call events
100  CONTINUE
c  Update statistics after all reps of nth engagement.
  DO 50 i=1,4
    istatb(i) = 0.5 + 100*statb(i) / nreps
    statb(i+4) = statb(i+4) / nreps
50  CONTINUE
2  format(f5.0,f6.3,f6.3,i4,i4,f6.2,f8.3,f8.3)
  exchc = 0
  if (statb(5).gt.0) exchc = statb(6)/statb(5)
  print 2,rg0,statb(5),statb(6),istatb(1),istatb(2),
1  exchc,statb(7), statb(8)
30  CONTINUE
END

c
SUBROUTINE FRD SSG (t, firer, tgt, armyf)
c  Frd ssg: Schedule effects after firing single shot gun.
  include 'meeting.h'
  logical can go
  integer armyf, firer, tgt, priorn
1  format('FRD SSG: t,firer,tgt,armyf=',f7.2,3i3)
2  format(f8.2, 1x, a4, i3, 'is out of ammo. Will attempt',
1  ' to hide if mobile.')
c
  if (trace) print *, '>frd ssg'
  IF (nrd(firer).lt.nrds(armyf)) THEN
c  Have ammo branch
c  change to put fot decision in priort HLReed 2-18-90
  IF(priorn(t,firer,level) .ne. tgt) THEN
c  switch tatgets
    busy(firer) = .false.
    call dis eng (t, firer, tgt,.true.,.true.)
c  If no other tgt and can move, skedul acceleration
    if (can go(firer,t) .and. ishtfs(armyf).eq.1)
1  call skedul(t,firer,'accel ',NULL)
    nrot(firer) = 0
  ELSEIF (tgt.gt.0) THEN
c  Schedule next round fired
    timea = tmin(armyf)
    timeb = tfixd(armyf,nrg)
    timec = tmedin(armyf) * exp(rolln(0.5))
    dt = amax1(timea,timeb+timec)
    call skedul (t+dt,firer,'fire ',tgt)
  ENDIF
  ELSE
c  Out-of-ammo branch

```

```

empty(firer) = .true.
IF (cango(firer,t)) THEN
  call skedul (t,firer,'accel ',NULL)
  call skedul (t+thide(armyf),firer,'hide ',NULL)
ENDIF
ENDIF
if (trace) print *, '<frd ssg'
END

c
SUBROUTINE HALT (t, firer)
c
Halt: simulate tank halting.
include 'meeting.h'
logical cango, threat
integer armyf, firer, tgt
1  format (f8.2,1x,a4,i3,' halts',12x,'(x=',f8.1,' y=',f8.1,')')
2  format(' HALT:  firer, Oturrer, Ohull =', i3, 2f8.1)
3  format(' HALT: firer, tgt, armyf, nrg =', 4i3)
4  format(' HALT: rx1,rx2,tfirst,dt =',5f10.3)
5  format(' HALT: t, tlastx, dt =',5f10.3)
c
if (trace) print *, '>halt'
if(invisb.eq.1)call cancel (firer,'vanish',NULL)
  narmy = armyf(firer)
call path (firer,t,motion(firer),0.0,x,y,vx,vy)
if (keyd(1).ge.2) print 1, t, color(narmy), firer, x, y
motion(firer) = STATNY
tlastx = t-3.
armyf = narmy
c
see if fire is a halt-to-fire-system and can still shoot
IF (ishtfs(armyf).eq.1 .and.
1  life(firer).lt.FKILL .AND. nrd(firer).lt.nrds(armyf)) THEN
c
This is a halt-to-fire system. schedule firing if tgt
c
still available.
threat = .false.
IF (nrtgt(firer).eq.FLS TGT) THEN
  threat = knceal(firer).ne.FD
ELSEIF (nrtgt(firer).gt.0) THEN
  threat = fot(firer,nrtgt(firer))
ENDIF
IF (.not.threat) THEN
c
  firer's tgt has vanished. firer may move
  if(cango(firer,t))call skedul (t, firer, 'accel ', NULL)
ELSE
  if (keyd(1).ge.2) print *, 'HALT: tlastx, aspect needs wk!'
  rx1=rolln(0.5)
  rx2=exp(rx1)
  tgt=nrtgt(firer)
  dummy = rgf(t,firer,tgt)
  dt = tfirst(armyf,nrg)*rx2
c change Dec 89 by HLReed

```

```

        dt = amax1(dt,tfire2(firer)+tmin(armyf) -t)
        nrib(firer) = 0
        nrot(firer) = 0
        call skedul (t+dt, firer, 'fire ', tgt)
    ENDIF
ENDIF
if (trace) print *, '<halt'
END

c
SUBROUTINE HIDE (t, tgt)
c
    Hide: Simulate tank hiding.
    include 'meeting.h'
    integer firer, tgt
1    format (f8.2,x,a4,i3,' goes into full defilade.')
c
    if (trace) print *, '>hide '
    if (keyd(1).gt.1) print 1, t, color(army(tgt)), tgt
    knceal(tgt) = FD
c
    Cancel all activities involving this tgt
c
    except discard rounds-in-flight in the impact routine
    firer = 1
    if (tgt.le.nblu) firer=nblu+1
    last = nblu
    if (tgt.le.nblu) last=nblu+nred
    DO 20 i=firer,last
        los(i,tgt) = .false.
        los(tgt,i) = .false.
20    CONTINUE
    call newtgt (t, firer, tgt)
    call cancel (tgt,'all ',NULL)
    call skedul(t,tgt,'slowup',NULL)
    call deaths(t)
    if (trace) print *, '<hide '
    END

c
SUBROUTINE IMPACT (t, ifirer, itgt)
c
    Impact: find what bullet does & what firer does.
c
    Changed for meeting program
    include 'meeting.h'
    logical hit
    integer expose

    if (trace) print *, '>impact'
c
    Find useful variables.
        n = army(ifirer)
        k = kindrd(n)
        expose = knceal(itgt)
        rgx = 0.0
c
    Find what bullet does.
        IF (itgt.eq.FLS TGT) THEN

```

```

c      Round does nothing.
      kshot(n,4) = kshot(n,4)+1
      ELSEIF (expose.eq.FD .and. k.le.2) THEN
c      Count round hitting berm.
      kshot(n,5) = kshot(n,5)+1
      if (keyd(1).ge.2) print *, 'Tgt in full defilade.'
      ELSE
c      See if round hits.
      call mayhit(t,ifirer,itgt,n,k,expose,hit)
      ENDIF
c      Find what firer does.
      IF (itgt.eq.FLS TGT .or. hit.and.tactic(n).eq.2 .or.
1      rgx.gt.4000.0) THEN
c      Switch targets if false target or rd hit & I switch on a hit.
c      Won't go here if I hit the berm; fls tgts don't go behind the
c      berm, and if true tgts do, the rd won't hit.
      ndet(ifirer) = ndet(ifirer)-1
      nrtgt(ifirer)=0
      call diseng(t,ifirer,itgt,.true.,.true.)
      ENDIF
      if (trace) print *, '<impact'
      END

c
      INTEGER FUNCTION !NDEXX(a, n, x)
c      -----
c      Find the index j, where a(j) <= x < a(j+1)
c      Adapted from Numerical Recipes, p90. The array ii must be increasing.
      integer n ,jl, ju, jm
      logical incre, above
      real a(n), x

c
      incre = a(n).gt.a(1)
      jl=0
      ju=n+1
10      IF (ju-jl.gt.1) THEN
          jm=(ju+jl)/2
          above=x.gt.a(jm)
          IF ((incre.and.above) .or. .not.(incre.or.above)) THEN
              jl=jm
          ELSE
              ju=jm
          ENDIF
      GOTO 10
      ENDIF
      indexx=jl
      END

c
      SUBROUTINE INIT
c      Init: Initialize scenario & schedule search at time zero.
c      Changed for meeting program

```

```

include 'meeting.h'
integer firer, tgt
logical regard
common /cregrd/ regard(NN)

c
if (trace) print *, '>Init'
call skedul(tmax,0,'finish',NULL)
call deploy
last = nred+nblu
call init2 (1, nblu)
call init2 (nblu+1, last)
c Set state variables for both red and blue systems.
DO 30 firer=1,last
  iflash(firer) = 0
  busy(firer) = .false.
  empty(firer) = .false.
  serchg(firer) = .true.
  ndet(firer) = 0
  regard(firer) = .true.
  manuvr(firer) = .false.
  DO 20 tgt=1,last
    foes(firer,tgt)= army(firer).ne.army(tgt)
    know(firer,tgt) = 0
    los(firer,tgt) = foes(firer,tgt) .and. invisb.ne.2
    fot(firer,tgt) = .false.
    seen(firer,tgt) = .false.
20  CONTINUE
30  CONTINUE
call serch1
if (trace) print *, '<Init'
END

c
SUBROUTINE INIT2 (ifirst, last)
c Init2: initialize each tank on one side.
c changed for meeting program
include 'meeting.h'
1 format(' INIT2: neval, nrd(1-3)=',4i5)
c
if (trace) print *, '>init2'
narmy = BLU
if (ifirst.gt.1) narmy=RED
last2 = nblu+nred
DO 10 i=ifirst, last
  army(i) = narmy
  life(i) = ALIVE
  nrd(i) = 0
  nrtgt(i) = 0
  nrot(i) = 0
  ichg(i) = 0
  IF(role(i).eq.ATTACK) THEN

```

```

        motion(i) = MAXVL
        knceal(i) = FE
    ELSE
        motion(i) = STATNY
        knceal(i) = HD
    ENDIF
    nhot(i) = 0
    DO 8 j=1,5
        msl fly(narmy,i,j) = 0
        nstats(j,narmy) = 0
8      CONTINUE
    DO 10 j=1,last2
        tfire(i,j) = 0.0
        tfire2(i) = - tmin(narmy)
        kncels(i,j) = .false.
10     CONTINUE
20     CONTINUE
    call detrg(narmy)
    call terrain (narmy,ifirst,last)
    if (trace) print *, '<init2'
    END

c
SUBROUTINE INPUT
c  Input: read misc inputs.
c  changed for meeting program
    include 'meeting.h'
    character*32 fname
    integer indx(5)
1    format(i1,a32)
4    format(a32)
c    read numbers of blue attackers, blue defenders,
c    red attackers, and red defenders, provision for decoys etc.
c    can be added later if desired
    read(5,*) nbatt,nbdef
    read(5,*) nratt,nrdef
    nblu = nbatt + nbdef
    DO 30 i = 1,nbatt
        sense(i) = -1.0
        role(i) = ATTACK
30    CONTINUE
    DO 40 i = nbatt + 1, nblu
        sense(i) = -1.0
        role(i) = DEFEND
40    CONTINUE
    nred = nratt + nrdef
    DO 50 i = nblu + 1, nblu + nratt
        sense(i) = +1.0
        role(i) = ATTACK
50    CONTINUE
    DO 60 i = nblu + nratt + 1, nblu + nred

```

```

        sense(i) = +1.0
        role(i) = DEFEND
60    CONTINUE
    read(5,*)(keyd(i),i=1,5)
    trace=keyd(4).gt.0
    read(5,*)indx
    DO 20 i=1,5
        if (indx(i).gt.1 .and. indx(i).le.20) keym(indx(i))=1
20    CONTINUE
    read(5,*)min rg, max rg, inc rg, irginc
    rgincr = irginc
    rginc2 = 0.5*irginc
    read(5,*) nreps, nwaves, iangd, meth sm, irandm
    read(5,*) tmax
    read(5,4) fname
    call rdmisc(fname,BLU)
c    Read pkh data for Blue.
        read 1, ipkh, fname
        call rdpkh(fname,BLU)
    read(5,4) fname
    call rdmisc(fname,RED)
c    Read pkh data for Red.
        read 1, ipkh, fname
        call rdpkh(fname,RED)
    read(5,*) invisb,n
    IF (invisb.ne.1) THEN
        print *, 'Smoke not played.'
    ENDIF
c    change by HL Reed to add new priority scheme 2-11-90
    read(5,4) fname
    open(4, file = fname, status = 'old')
    rewind(4)
    DO 100 n = 1, 192
        read(4,*) n1,n2,n3,n4,n5,n6, lpri(n1,n2,n3,n4,n5,n6,1)
100    CONTINUE
    close(4)
    read(5,4) fname
    open(4, file = fname, status = 'old')
    rewind(4)
    DO 200 n = 1, 192
        read(4,*) n1,n2,n3,n4,n5,n6, lpri(n1,n2,n3,n4,n5,n6,2)
200    CONTINUE
    close(4)
    if (trace) print *, '<input'
    END

c
SUBROUTINE KILL (firer, tgt, hit, injury,r)
c    Kill: find kill type for a hit on a tgt.
c    The routine is called by mayhit.
c    changed May 18,1989 for simplified hit and kill model, HL Reed

```

```

include 'meeting.h'
logical hit
integer firer, tgt
common /cpkh2/ pkill(2,9,2,5,9)
save /cpkh2/
c Change for interpolation on range for pkill
  p(i) = (1.0 - r) * pkill(narmy,ncase,nhdfe,i,jrg)
  1      + r * pkill(narmy,ncase,nhdfe,i,jrg+1)
  if (trace) print *, '>kill'
  nhdfe = knceal(tgt)-1
  narmy = army(firer)
  IF (motion(firer) .eq. STATNY) THEN
    ncase = 0
  ELSE IF (manuvr(firer)) THEN
    ncase = 6
  ELSE
    ncase = 3
  END IF
  IF (motion(tgt) .eq. STATNY) THEN
    ncase = ncase + 1
  ELSE IF (manuvr(tgt)) THEN
    ncase = ncase + 3
  ELSE
    ncase = ncase + 2
  END IF
c Find kill level
c Change 12-9-89 by HLReed for interpolation on range for pkill
c Get ratio based on 500 meter intervals
  r = r/500.
c Is range > 4000 meters if so then use 3999.5
  if(r .GE. 8.) r = 7.999
c Get integer part
  jrg = int(r)
c and fractional part
  r = r - float(jrg)
c Correct for the fact that indices start at 1 rather than 0
  jrg = jrg + 1
  temp = ranu(0.0)
  IF (temp .gt. p(1)) THEN
c no hit and no kill
  hit = .false.
  injury = ALIVE
  ELSEIF (temp .gt. p(2)) THEN
c a hit and a "k" kill
  hit = .true.
  injury = KKILL
  ELSEIF (temp .gt. p(3)) THEN
c a hit and an "m&f" kill but no "k"
  hit = .true.
  injury = MFKILL

```

```

        ELSEIF (temp .gt. p(4)) THEN
c a hit and an "f" kill but no "m" kill
        hit = .true.
        injury = FKILL
        ELSEIF (temp .gt. p(5)) THEN
c a hit and an "m" kill but no "f" kill
        hit = .true.
        injury = MKILL
        ELSE
c a hit but no kill
        hit = .true.
        injury = ALIVE
        ENDIF
        if (injury.eq.ALIVE) kshot(narmy,10) = kshot(narmy,10)+1
        if (injury.eq.MKILL) kshot(narmy,11) = kshot(narmy,11)+1
        if (injury.eq.FKILL) kshot(narmy,12) = kshot(narmy,12)+1
        if (injury.eq.MFKILL) kshot(narmy,13) = kshot(narmy,13)+1
        if (injury.eq.KKILL) kshot(narmy,14) = kshot(narmy,14)+1
        if (trace) print *, '<kill'
        END
c
        SUBROUTINE LATE KL (t, tgt,ji)
c Late kl: Simulate recognition of m&f kill after period of inactivity.
        include 'meeting.h'
        integer firer, tgt
1      format(f8.2,1x,a4,i3,' l-killed.')
c
        if (trace) print *, '>latekl'
        if (keyd(1).gt.1) print 1, t, color(army(tgt)), tgt
        firer = 1
        if (tgt.le.nblu) firer=nblu+1
        life(tgt) = IKILL
        call cancel (tgt, 'ikill ',NULL)
        call newtgt (t,firer,tgt)
        call deaths(t)
        if (trace) print *, '<latekl'
        END
c
        SUBROUTINE MAX VEL(t, firer)
c Max vel: simulate tank reaching cruise speed.
        include 'meeting.h'
        integer firer, tgt
1      format (f8.2,1x,a4,i3,' at full speed.')
c
        if (trace) print *, '>maxvel'
        if (keyd(1).ge.2) print 1, t, color(army(firer)), firer
        call path(firer,t,motion(firer),0.0,x,y,vx,vy)
        motion(firer) = MAXVL
        tgt = nrtgt(firer)
        IF (tgt.gt.0) THEN

```

```

    if (life(tgt).lt.IKILL) call engage(t,t,firer,nrtgt(firer))
ENDIF
if (trace) print *, '<maxvel'
END

```

c

```

SUBROUTINE MAYHIT (t,l,it,n,k,expose,hit)

```

c

```

Mayhit: Find what the round does.

```

c

```

Changed May 18, 1989 for simplified hit and kill model, HL Reed
include 'meeting.h'
logical hit
integer expose

```

```

if (trace) print *, '>mayhit'
kshot(n,6) = kshot(n,6)+1

```

c

```

Find whether a hit occurs.

```

```

hit = .false.
rgx = rgf(t,l,it)
r = rgx
call kill(l,it,hit,injury,r)

```

```

IF (hit) THEN

```

c

```

Treat hit.

```

```

kshot(n,8) = kshot(n,8)+1
if (life(it).eq.MFKILL) nhot(it)=nhot(it)+1
if (nhot(it).gt.nbump(n)) call skedul(t,it,'ikill ',NULL)
know(l,it)=2
IF (reliab(n) .ge. ranu(0)) THEN
    call damage(t, l, it, injury)
ELSE

```

c

```

Round is a dud.

```

```

kshot(n,9) = kshot(n,9)+1
ENDIF

```

```

ELSE

```

c

```

Treat miss.

```

```

kshot(n,7) = kshot(n,7)+1
IF (psense(n,nrgf(rgx,rgincr)) .gt.ranu(0.0)) THEN
    know(l,it)=1
    if (keyd(1).ge.2) print *, 'Miss is sensed.'
ELSE
    if (keyd(1).ge.2) print *, 'Miss is not sensed.'
ENDIF

```

```

ENDIF

```

```

END

```

c

```

SUBROUTINE NEWTGT (t, firer, tgt)

```

c

```

New tgt: redirect all 'attackers' of tgt to a new target.

```

c

```

New tgt called for non-false tgts only and only if tgt condition
warrants it. It should only be called if tgt is V-killed,

```

c

```

vanishes, or hides.

```

c

```

Maybe it should be called if the tgt is I-killed by a gun system.
include 'meeting.h'

```

```

integer first, firer, tgt, armyf, armyt
logical hav amo, cango
1  format(f8.2,1x,a4,i3,' dis-engs ',a4,i3,20x,'#tgts=',i2)
2  format(f8.2, 1x, a4, i3, ' begins to reload.')
c
if (trace) print *, '>newtgt'
c  Find first and last 'attacker'
    first = 1
    if (firer.gt.nblu) first = nblu+1
    last = nblu
    if (firer.gt.nblu) last = nblu+nred
    armyf = army(first)
    armyt = 3-armyf
    kind = kindrd(armyf)
    nrpb2 = nrpb(armyf)
    DO 20 j=first, last
        IF ((fot(j,tgt)) .and. life(j).lt.FKILL) THEN
c          Single shot gun system or other fire & forget system.
c          Single shot gun system.
            call cancel(j,'fire ',tgt)
            if (nrtgt(j).eq.tgt) busy(j) = .false.
            if (nrtgt(j).eq.tgt) nrtgt(j) = 0
            hav amo = nrd(j).lt.nrds(armyf)
            IF (hav amo) THEN
                thuman = 0.*exp(rolln(0.5))
                call selecs(t,j,thuman)
            ELSEIF (can go(j,t).and.ishtfs(armyf).gt.0) THEN
c              Move out
                call skedul(t,j,'accel ',NULL)
            ENDIF
            nrot(j) = 0
            fot(j,tgt) = .false.
            if (keyd(1).ge.2) print 1, t, color(armyf), j,
1              color(armyf), tgt, ndummy
            nrtgt(j) = 0
            fot(j,tgt) = .false.
        ENDIF
        if (seen(j,tgt)) ndet(j) = ndet(j) - 1
        seen(j,tgt) = .false.
20    CONTINUE
    IF (.not.repeat) THEN
        repeat = .true.
        call skedul (t+.01,0,'search',NULL)
    ENDIF
    if (trace) print *, '<newtgt'
    END
c
c  SUBROUTINE NEXTEV (l,act,it,t)
c  Nextev: Find the next scheduled event.
include 'clock.h'

```

```

character*6 act
c
c  Fill arguments
  l = who(nxevnt)
  act = what(nxevnt)
  it = whom(nxevnt)
  t = when(nxevnt)
c  Drop storage unit from active storage chain
  n = nxevnt
  nxevnt = next(nxevnt)
c  Add storage unit to inactive storage.
  next(n) = nxidle
  nxidle = n
END
c
c  FUNCTION NRGF (rg,rgincr)
c  Nrgf: find which rgincr meter rg band range is in.
  nrgf = max0(1,int(0.5+rg/rgincr))
END

SUBROUTINE PATH (firer,t, motio2, delt, x, y, vx, vy)
c  Path: search path table for position and vel at time t.
c  Changed for meeting program to allow for positive or negative
c  velocities for the vehicles
  include 'meeting.h'
  logical is atkr, kan go, old
  integer firer
c
  if (trace) print *, '>path'
  is atkr = role(firer) .eq. ATTACK
  kan go = (motio2.ne.STATNY .or.
1  life(firer).eq.ALIVE .or. life(firer).eq.FKILL)
  dt = t-t0(firer)
  old = dt .gt. delt
  IF (is atkr .and. kan go .and. old) THEN
c  Update positions and velocity.
    t0(firer) = t
    if (motio2.eq.SLOWNG) THEN
      dv = sense(firer)*decel(firer)*dt
      y0(firer) = y0(firer)+dt*(vy0(firer)-0.5*dv)
      v = vy0(firer)-dv
      if (abs(v).lt.0.001) v = 0.0
      vy0(firer) = v
    ELSEIF (motio2.eq.STATNY) THEN
      vy0(firer) = 0.0
    ELSEIF (motio2.eq.ACCELG) THEN
      dv = sense(firer)*accel(firer)*dt
      y0(firer) = y0(firer)+dt*(vy0(firer)+0.5*dv)
      vy0(firer) = vy0(firer)+dv
    ELSEIF (motio2.eq.MAXVL) THEN

```

```

        y0(firer) = y0(firer)+vy0(firer)*dt
        vy0(firer) = sense(firer)*speed(firer)
    ELSE
        print *, 'PATH: no such motion. motio2=,', motio2
    STOP
    ENDIF
ENDIF
x=x0(firer)
y=y0(firer)
vy=vy0(firer)
vx=0.0
IF(army(firer).eq.BLU) THEN
    if(y.lt.100.0) call skedul(t+0.,NULL,'finish',NULL)
ELSE
    if(y.gt.(rg0-100.)) call skedul(t+0.,NULL,'finish',NULL)
ENDIF
if (trace) print *, '<path'
END

c
SUBROUTINE PINPNT (t,firer)
c   Pinpnt: Simulate firing signature (pinpoint) detection by some foes.
include 'meeting.h'
integer first, firer
logical wilsee
1   format (f8.2,1x,a4,i3,' sees   ',a4,i3,' muzzle flash')
c
    if (trace) print *, '>pinpnt'
    first = 1
    if (firer.le.nblu) first = nblu+1
    last = nblu
    if (firer.le.nblu) last = nblu+nred
    pinpxx = pinp(army(first))
    DO 20 i=first, last
        wilsee = pinpxx.gt.ranu(0.0)
        IF (life(i).lt.FKILL .and. wilsee .and.
2       ndet(i).lt.ndets(army(i)) .and.
1       los(i,firer) .and. .not.seen(i,firer)) THEN
        if (keyd(1).ge.2) print 1,
1       t, color(army(i)), i, color(army(firer)), firer
        seen(i,firer) = .true.
        ndet(i) = ndet(i) + 1
        thuman = pntime(army(i)) *exp(rolln(0.5))
        call selects(t,i,thuman)
        if(xxfer(army(i))) call skedul(t+thuman,first,'xfer ',firer)
    ENDIF
20  CONTINUE
    if (trace) print *, '<pinpnt'
END

c
INTEGER FUNCTION PRIORN (t, firer, lev old)

```

```

c   Priorn: select tgt with highest priority.
c   changed for new priority model
    include 'meeting.h'
    logical better, ck tgt
    integer firer, armyf

c
    if (trace) print *, '>priorn'
    armyf = army(firer)
c   'make' dummy tgt for comparison
    rg old=1.e35
    t old=1.e35
    lev old=1000
    priorn = NULL
    last = nbnu+nred
    DO 30 mtgt=1,last
c   Compare all possible targets
c   Change HL REED 2-18-90
    rg tgt = rgf (t,firer,mtgt)
    ck tgt = seen(firer,mtgt) .and. life(mtgt).lt.IKILL
1   .and. rgtgt.le.4000.0
    IF (ck tgt) THEN
c   Firer sees tgt, it's threatening, & he's not firing at it.
    call priort(firer, mtgt, rg tgt, t, level)
c   Now pick the tgt with highest priority
    rg tgt = rg tgt *(1+.05*rolln(1.0))
    t tgt = tfire(firer,mtgt)
    better = level .lt. lev old
    IF (lev old.eq.level) THEN
c   Same priority class; now break ties
c   if new tgts pick closer
    if (t tgt.le. 0) better = rg tgt .lt. rg old
c   if old tgts, pick older (least recently fired on)
    if (t tgt.gt. 0) better = t tgt .lt. t old
    ENDIF
    IF (better) THEN
        lev old = level
        t old = t tgt
        rg old = rg tgt
        priorn = mtgt
    ENDIF
    ENDIF
30  CONTINUE
    if (trace) print *, '<priorn'
    END

c
    SUBROUTINE PRIORT(firer, tgt, rg tgt, t, L)
c   PRIORT: find priority of tgt
c   Changed by HLReed 2-8-90
    include 'meeting.h'
    integer firer, tgt

```

```

1  format(' PRIORT: ',a4,i3,' consids ',a4,i3,' with priority',
1  i4,' (',i2,')')
  if (trace) print *, '>priorit'
  j = army(firer)
  IF(fot(firer,tgt)) THEN
    n7 = 1
  ELSE
    n7 = 2
  ENDIF
  n6 = 1
  do 20 jjj = 1, nblu+nred
    if(fot(jjj,tgt) .and. (jjj .ne. firer)) n6 = 2
20  continue
  n6 = n6 + n6 + n7 - 2
  IF (tfire(firer,tgt).le.0) THEN
c   new target
    n5 = 3
  ELSE
    n5 = 2
    if(know(firer,tgt) .eq. 1) n5 = 1
  ENDIF
  IF (rg tgt.lt.recknz(army(firer))) THEN
    n4 = 1
  ELSE
    n4 = 2
  ENDIF
  t activ = 1.e35
  if (tfire2(tgt).gt.0.) t activ = t-tfire2(tgt)
  IF (t activ .lt. 30.) THEN
    n3 = 1
  ELSE
    n3 = 2
  ENDIF
  IF (nrtgt(tgt).ne.0) THEN
c   target has a target
    n2 = 1
  ELSE
    n2 = 2
  ENDIF
  m = motion(tgt)
  IF((m.eq.STATNY) .or. (m.eq.SLOWNG .and. n4.eq.1)) THEN
    n1 = 1
  ELSE
    n1 = 2
  ENDIF
  L = lpri(n1,n2,n3,n4,n5,n6,j)
  if (trace) print *, '<priorit'
  END
c
  FUNCTION RANU (dm)

```

c Ranu: A version of uran31 random uniform nr generator.

```
common /crandm/ i, j
real a1
j=i
j=j*25
j=j-(j/67108864)*67108864
j=j*25
j=j-(j/67108864)*67108864
j=j*5
j=j-(j/67108864)*67108864
a1=j
i=j
ranu= a1/67108864
END
```

c

SUBROUTINE RD MISC (dbname,narmy)

c Rd misc: read miscellaneous tank characteristics.

c changed for new meeting model

include 'meeting.h'

character dbname*32

2 format(1a1)

3 format(a)

c

if (trace) print *, '>Rdmisc'

open(4, file=dbname, status='old')

rewind 4

read(4,*) (sysdim(narmy,i),i=1,8)

read(4,*) (psense(narmy,i),i=1,8)

c Read nvl outputs.

read(4,*) (pinfin(narmy,1,j),j=1,8)

read(4,*) (pinfin(narmy,2,j),j=1,8)

read(4,*) (pinfin(narmy,3,j),j=1,8)

read(4,*) (tbar(narmy,1,j),j=1,8)

read(4,*) (tbar(narmy,2,j),j=1,8)

read(4,*) (tbar(narmy,3,j),j=1,8)

read(4,*) (recknz(narmy),(pfalse(narmy,i),i=1,2),

1 tlook(narmy),pinp(narmy),reliab(narmy),trelod(narmy),

2 pntime(narmy)

read(4,*) (nrds(narmy),nrpt(narmy),nrpb(narmy),

1 tactic(narmy),kind rd(narmy),nprior(narmy),

2 ndummy,ndummy

read(4,*) (tof(narmy,i),i=1,8)

read(4,*) (tfirst(narmy,i),i=1,8)

read(4,*) (tmedin(narmy), tmin(narmy), rof(narmy)

read(4,*) (tfixed(narmy,i),i=1,8)

IF(narmy.eq.BLU) THEN

 i = 1

 j = nblu

ELSE

 i = nblu + 1

```

        j = nblu + nred
    ENDIF
    read(4,*) accel(i), decel(i), speed(i),
1    angle(narmy), thide(narmy)
    DO 100 i1 = i+1,j
        accel(i1) = accel(i)
        decel(i1) = decel(i)
        speed(i1) = speed(i)
100    CONTINUE
    read(4,*) ishtfs(narmy), nbump(narmy), ibump
    tbump(narmy) = ibump
c    eventually should remove references to decoys here and add that
c    reference to code above
    read(4,*) ndecoy(narmy), nflash(narmy)
c    eventually should remove reference to share (not needed for new
c    priority model-but would have to change a lot of existing input data)
    read(4,*) share(narmy), xxfer(narmy)
    read(4,2) kview(narmy)
    read(4,*) ndets(narmy)
    close (4)
c    if (keyd(2).gt.0) call pr misc (narmy)
c    Convert tbar to detection probability / second.
        DO 30 i=1,8
            DO 20 j=1,3
                tbar(narmy,j,i) = 1.0-exp(-1.0/tbar(narmy,j,i))
20            CONTINUE
30        CONTINUE
    if (trace) print *, '<rdmisc'
    END

c
    SUBROUTINE RDPKH (dbname, narmy)
c    Rd pkh: read probability-of-kill data.
c    Changed for simplified hit and kill model May 19,1989, HL Reed
    include 'meeting.h'
    character*32 dbname
    common /cpkh2/ pkill(2,9,2,5,9)
    save /cpkh2/
    if (trace) write(*,*)>rdpkh'
    open (4, file=dbname, status='old')
    rewind 4
    DO 100 ncase = 1,9
    DO 70 nhdfc=1,2
    DO 30 i=1,5
        read (4,*) n1,n2,n3,(pkill(narmy,ncase,nhdfc,i,j),j=1,9)
30    CONTINUE
70    CONTINUE
100    CONTINUE
    close(4)
90    if (trace) write(*,*)>rdpkh'
    END

```

```

c      SUBROUTINE RESET (prflg)
c      Reset: Initialize the clock to time zero.
      include 'clock.h'
      logical prflg

c      prflag = prflg
      nxevnt = 0
      nxidle = 1
      DO 10 j=1,NE
        next(j) = j+1
10     CONTINUE
        next(NE) = 0
      END

c      FUNCTION RGF (t, firer, tgt)
c      Rgf: find the position of the firer w.r.t. the tgt.
      include 'meeting.h'
      integer firer, tgt
      common /pathc / xf, yf, xt, yt
      save /pathc /
1      format (9x,'Firer  x, y, vx, vy =', 4f10.1, /
*         9x,'Target x, y, vx, vy =', 4f10.1)
c
      if (trace) print *, '>rgf'
      call path (firer,t,motion(firer),0.0,xf,yf,vf(1),vf(2))
      call path (tgt,t,motion(tgt),0.0,xt,yt,vt(1),vt(2))
      s(1) = xf-xt
      s(2) = yf-yt
      s(3) = 0.0
      vt(3) = 0.0
      vf(3) = 0.0
      temp = vabs(s)
      if(temp.GT.4000) temp = 4000
      nrg = nrgf(temp,rgincr)
      rgf = temp
      rg = irginc*nrg
      if (keym(20).gt.0) print 1,
*      xf, yf, vf(1), vf(2), xt, yt, vt(1), vt(2)
      if (trace) print *, '<rgf'
      END

c      FUNCTION ROLLN(sigma)
c      Rolln: find a random number from a normal distribution.
c      Box-Muller method
      save j, z
      data j/0/

c      IF (j.eq.0) THEN
        x = sqrt(-2.*alog(ranu(dm)))

```

```

    y = 2.*3.1415926535*ranu(dm)
    rolln = x*cos(y)*sigma
    z = x*sin(y)
ELSE
    j = 1-j
    rolln = z*sigma
ENDIF
END

c
SUBROUTINE SEARC2 (t,firer,tgt,narmy,cond,dt)
c
  Searc2: see if a tank detects a target during this second.
  include 'meeting.h'
  integer firer, tgt, cond
c
  if (trace) print *, '>searc2'
  temp = rg/rgincr
  indx = int(temp)
  IF (indx .lt. 1) THEN
    tlo = 1.0
    thi = tbar(narmy,cond,1)
  ELSEIF (indx .lt. 8) THEN
    tlo = tbar(narmy,cond,indx)
    thi = tbar(narmy,cond,indx+1)
  ELSE
    tlo = tbar(narmy,cond,8)
    thi = 0.0
  ENDIF
  frac = temp-aint(temp)
  pdetct = tlo + frac*(thi-tlo)
  IF (ranu(0.0).gt.pdetct) THEN
c
    The firer doesn't detect the target in the next second.
    repeat = .true.
    dt = 1.0
  ELSE
c
    This firer detects the target in this second.
    call skedul(t+ranu(0.0),firer,'detect', tgt)
  ENDIF
  if (trace) print *, '<searc2'
  END

c
SUBROUTINE SEARCH (t)
c
  Search: see if any targets are detected in1 the next second.
  include 'meeting.h'
  logical ignore
  common /cserch/ i1,in,j1,jn,rgtbl(NN,NN),ignore(NN),
1  ymax(NN),iarmy,jarmy,ndeti,ndetj
  save /cserch/
  rss(x,y) = sqrt(x*x+y*y)
c
  if (trace) print *, '>search'

```

```

repeat = .false.
c   Update status of tanks.
c   (Next line shud eventually be updated in damage.f, ltkill.)
DO 11 i = 1, nblu + nred
  ignore(i) = ignore(i).or.life(i).ge.IKILL
11  CONTINUE
DO 20 i=i1,in
  IF (.not.ignore(i) ) THEN
    call path(i,t,motion(i),0.0,dm,dm,dm,dm)
    DO 10 j=j1,jn
      IF (.not.ignore(j) ) THEN
        call path(j,t,motion(j),0.0,dm,dm,dm,dm)
        rgtbl(i,j) = rss(x0(i)-x0(j),y0(i)-y0(j))
        rgtbl(j,i) = rgtbl(i,j)
      ENDIF
    CONTINUE
  ENDIF
10  CONTINUE
20  CONTINUE
c
DO 40 i=i1,in
c   Loop thru Southern tanks.
  IF (.not.ignore(i)) THEN
c   Consider tank i (It is alive and can detect or be detected.)
c   Change made March 20, 1989 by H.L.Reed to allow the individual condition
c   of each target tank to be used to define the probability of acquisition
    icond = 2
    if(motion(i).ne.STATNY) icond = 3
    if(knceal(i).eq.HD) icond = 1
    DO 30 j=j1,jn
      IF (.not.ignore(j)) THEN
c   Consider tank j (Also alive and can detect or be detected.)
        jcond = 2
        if(motion(j).ne.STATNY) jcond = 3
        if(knceal(j).eq.HD) jcond = 1
        rgi = rgvis(jcond,i)
        rgj = rgvis(icond,j)
        rgmax = amax1(rgi,rgj)
        rg = rgtbl(i,j)
        IF (rg.lt.rgmax) THEN
c   At least one is in detection rg of the other.
          IF (los(i,j)) THEN
c   Line-of-sight exists between them.
c   Treat Southern tank as searcher
            IF (rg.lt.rgi .and. .not.seen(i,j) .and.
1          ndet(i).lt.ndeti) THEN
              call searc2(t,i,j,iarmy,jcond,dt)
            ELSE
              repeat = .true.
            ENDIF
c   Treat Northern as searcher
          
```

```

1      IF (rg.lt.rgj .and. .not.seen(j,i) .and.
      ndet(j).lt.ndetj) THEN
      call searc2(t,j,i,jarmy,icond,dt)
      ELSE
      repeat = .true.
      ENDIF
      ELSE
      repeat = .true.
      ENDIF
      ENDIF
      ENDIF
30     CONTINUE
      ENDIF
40     CONTINUE
      if (repeat) call skedul(t+1.0,0,'search', NULL)
      if (trace) print *, '<search'
      END
c
      SUBROUTINE SELECS (t,firer,dt)
      include 'meeting.h'
      logical loaded
      integer firer, armyf
1      format (f8.2,1x,a4,i3,' does not select; selecting already.')
2      format (f8.2,1x,a4,i3,' does not select; channels full.')
3      format (f8.2,1x,a4,i3,' does not select; pod empty.')
4      format (f8.2,1x,a4,i3,' begins selection.')
c
      if (trace) print *, '>selecs'
      armyf = army(firer)
      loaded = nrtgt(firer).ne.0
      IF (busy(firer) .or. empty(firer) .or. loaded) THEN
c      Wait cause busy selecting, pod empty, or channels full.
      IF (keyd(1).ge.2) THEN
      IF (busy(firer)) THEN
      print 1, t, color(armyf), firer
      ELSEIF (loaded) THEN
      print 2, t, color(armyf), firer
      ELSEIF (empty(firer)) THEN
      print 3, t, color(armyf), firer
      ENDIF
      ENDIF
      ELSE
c      Start selection: none in progress and a channel is free.
      busy(firer) = .true.
      call skedul(t+dt,firer,'select', NULL)
      if (keyd(1).ge.2) print 4, t, color(armyf), firer
      ENDIF
      if (trace) print *, '<selecs'
      END
c

```

```

SUBROUTINE SELECT (t, firer)
c   Select: gunner chooses most dangerous target he sees.
   include 'meeting.h'
   character*4 colort
   logical tgt fls, f alive, can go
   integer firer, tgt, priorn, armyf
1   format(f8.2,1x,a4,i3,' selects ',a4,i3,' with priority',i4,
1   ' #tgts=',i2)
2   format(f8.2,1x,a4,i3,' selects ',a4,' -1',
1   ' & discards ',a4,i3,' #tgts=',i2)
3   format(f8.2,1x,a4,i3,' selects',8x,'- (empty target set)')
4   format(' SELECT: ',a4,i3,' selects ',a4,i3,' with priority',i4)
c
   if (trace) print *, '>select'
   armyf = army(firer)
   kind = kindrd(armyf)
   f alive = life(firer).lt.FKILL
   IF (f alive) THEN
c   Firer can shoot, so have him select.
       tgt = priorn(t,firer,level)
       IF (tgt.eq.NULL) THEN
c   Firer has no targets to select so he moves if possible
           if (keyd(1).ge.2) print 3, t,color(armyf), firer
           busy(firer) = .false.
           IF (can go(firer,t)) THEN
               call cancel(firer,'halt ', NULL)
               call cancel(firer,'accel ', NULL)
               call skedul(t,firer,'accel ',NULL)
           ENDIF
       ELSE
c   Tgt has been selected
           colort = color(army(tgt))
           IF (tfire(firer,tgt).le.0.) THEN
c   Tgt is new; replace with false tgt randomly.
               i = knceal(tgt)-1
               pf = ranu(0)
               tgt fls = pf .lt. pfalse(armyf,i)
               IF (tgt fls) THEN
                   seen(firer,tgt) = .false.
                   if (keyd(1).ge.2) print 2, t, color(armyf),
1                   firer, colort, colort, tgt, ndummy
               tgt = FLS TGT
c   Restart search if it is turned off
                   IF (.not.repeat) THEN
                       repeat = .true.
                       call skedul(t,0,'search',NULL)
                   ENDIF
               ELSE
                   fot(firer,tgt) = .true.
                   if (keyd(1).ge.2) print 1, t, color(armyf),

```

```

1      firer,colort,tgt,level,ndummy
      ENDIF
      ELSE
c      Firer has previously serviced this target.
        fot(firer,tgt) = .true.
        if (keyd(1).ge.2) print 1, t, color(armyf),
1      firer,colort,tgt,level,ndummy
      ENDIF
        call engage (t, t, firer, tgt)
      ENDIF
        nrtgt(firer) = tgt
      ENDIF
        if (trace) print *, '<select'
      END

c
c      SUBROUTINE SERCH0
c      Serch0: Find useful constants for search.
        include 'meeting.h'
        logical ignore
        common /cserch/ i1,in,j1,jn,rgtbl(NN,NN),ignore(NN),
1      ymax(NN),iarmy,jarmy,ndeti,ndetj
        save /cserch/

        if (trace) print *, '>serch0'
c      Find 1st and last in Southern & Northern forces.
        i1 = nblu+1
        in = nblu+nred
        j1 = 1
        jn = nblu
        DO 20 i=1,nblu + nred
          ignore(i) = .false.
20      CONTINUE
        iarmy = army(i1)
        jarmy = army(j1)
        ndeti = ndets(iarmy)
        ndetj = ndets(jarmy)
        if (trace) print *, '<serch0'
      END

c
c      SUBROUTINE SERCH1
c      Find whether & when search should be started.
c      Changed to make situation symmetric for meeting program
        include 'meeting.h'
        logical ignore
        common /cserch/ i1,in,j1,jn,rgtbl(NN,NN),ignore(NN),
1      ymax(NN),iarmy,jarmy,ndeti,ndetj
        save /cserch/

        if (trace) print *, '>serch1'
        call serch0

```

```

dt = tmax + 1.0
c  Loop thru Southern force and Northern force.
DO 40 i=i1,in
  DO 30 j=j1,jn
    x = x0(j)-x0(i)
    y = y0(j)-y0(i)
    d = sqrt(x**2 + y**2)
    rgtbl(i,j) = d
    rgtbl(j,i) = d
    icond = 2
    if(motion(i).ne.STATNY) icond = 3
    if(knceal(i).eq.HD) icond = 1
    jcond = 2
    if(motion(j).ne.STATNY) jcond = 3
    if(knceal(j).eq.HD) jcond = 1
    r = amax1(rgvis(jcond,i),rgvis(icond,j))
    IF (r .gt. d) THEN
c    At least one is in detection range at time zero.
      dt = 0
    ELSE
c    Neither is in detection range at time zero.
      IF(role(i).eq.ATTACK .and. role(j).eq.ATTACK) THEN
        v = speed(i) + speed(j)
      ELSE IF(role(i).eq.ATTACK) THEN
        v = speed(i)
      ELSE IF(role(j).eq.ATTACK) THEN
        v = speed(j)
      ELSE
        v = 0.0
      ENDIF
      IF (v .gt. 0.0 .and. abs(x) .lt. r) THEN
c    At least one will enter.
        q = sqrt(r**2 - x**2)
        dt = amin1(dt,q/v)
      ENDIF
    ENDIF
  30  CONTINUE
  40  CONTINUE
  if (dt.lt.tmax) call skedul(dt,ALL,'search',ALL)
  repeat = dt.lt.tmax
  if (trace) print *, '<serch1'
  END
c
c  SUBROUTINE SKEDUL (t,l,act,it)
c  Schedule: Schedule an event for later execution.
  include 'clock.h'
  character*6 act
  1  format(9x,'skedul ',i3,' ',a6,i3,' at time',f8.2)
c
  if (prflag) print 1, l, act, it, t

```

```

IF (nxidle.eq.0) THEN
c  If storage all used stop
    print *, 'Storage overloaded with too many events.'
    STOP
ELSE
c  Store the event
c  Cut storage unit from empties
    n = nxidle
    nxidle = next(nxidle)
c  Then find where to insert this event in the event list.
    IF (nxevnt.le.0) THEN
c  New event is only event
        next(n) = 0
        nxevnt = n
    ELSE
c  Then find where to insert it.
c  Point to first 2 events
        l = nxevnt
        m = next(l)
c  Find where to insert them
        IF (t.ge.when(l)) THEN
c  See if between 2 scheduled events.
c  Loop till found.
20      IF (m.ne.0 .and. t.ge.when(m)) THEN
            l = m
            m = next(m)
            GOTO 20
        ELSE
c  Splice new event into list
            next(n) = m
            next(l) = n
        ENDIF
    ELSE
c  Place new event as most imminent
        next(n) = nxevnt
        nxevnt = n
    ENDIF
ENDIF
c  Finally store event info
    when(n) = t
    what(n) = act
    who(n) = l
    whom(n) = it
ENDIF
END

c
SUBROUTINE SLOW UP (t, firer)
c  Slow up: simulate tank starting to slow down.
c  Changed to allow motion in both direction for meeting program
include 'meeting.h'

```

```

integer firer
1  format (f8.2,1x,a4,i3,' continues to slow up.')
2  format (f8.2,1x,a4,i3,' would slow up if it weren't',
1  ' already stopped.')
3  format (f8.2,1x,a4,i3,' brakes',11x,'(was accelerating)')
4  format (f8.2,1x,a4,i3,' brakes',11x,'(was cruising)')
c
if (trace) print *, '>slowup'
kind mv = motion(firer)
narmy = army(firer)
IF (kind mv.eq.SLOWNG) THEN
c  Previous motion was slowing
  if(keyd(1).ge.2)print 1, t, color(narmy), firer
ELSE IF (kind mv.eq.STATNY) THEN
c  Previous motion was stationary
  if(keyd(1).ge.2)print 2, t, color(narmy), firer
ELSE IF (kind mv.eq.ACCELG) THEN
c  Previous motion was accelerating
  if(keyd(1).ge.2)print 3, t, color(narmy), firer
  call path (firer,t,motion(firer),0.0,x,y,vx,vy)
  dt =abs(vy)/decel(firer)
  motion(firer) = SLOWNG
  call skedul(t+dt,firer,'halt ', NULL)
ELSE IF (kind mv.eq.MAXVL) THEN
c  Previous motion was cruising at max vel
  if(keyd(1).ge.2)print 4, t, color(narmy), firer
  call path (firer,t,motion(firer),0.0,x,y,vx,vy)
c  schedule halt time
  dt = abs(vy)/decel(firer)
  call skedul(t+dt,firer,'halt ', NULL)
  motion(firer) = SLOWNG
ENDIF
if (trace) print *, '<slowup'
END

c
SUBROUTINE TERAIn (narmy,ifirst,last)
c  Mask st: find path lengths where attacker is masked by terrain
c  changed for meeting model to allow both sides to disappear
include 'meeting.h'
common /terane/ d(2,40), xold(20), yold(20), dist(20), iseg(20)
1  format (' visible for',f5.0,'m, then hidden for',f5.0,'m.')
c
if (trace) print *, '>terain'
c  Find segment length at start of each engagement.
DO 20 i=1,39,2
c  Hunfeld terrain constants
  f = -alog(ranu(0.0))
  d(narmy,i) = 300.*f**1.2
  f = -alog(ranu(0.0))
c  d(narmy,i+1) = 750.*f**2.0

```

```

        d(narmy,i+1) = 100.*f
        if (keyd(1).ge.2) print 1, d(narmy,i), d(narmy,i+1)
20    CONTINUE
c    Initialize data for each tank
    DO 30 i=first,last
        call path (i,0.,motion(i),0.0,x,y,vx,vy)
        xold(i) = x
        yold(i) = y
        dist(i) = d(narmy,1)
        iseg(i) = 1
        IF(role(i).eq. ATTACK) THEN
            call skedul (0.,i,'vanish',NULL)
        ENDIF
30    CONTINUE
    if (trace) print *, '<terain'
    END

c
    FUNCTION VABS (a)
c    Vabs: find absolute value of a vector (magnitude).
    dimension a(3)
    vabs = sqrt(a(1)**2 + a(2)**2 + a(3)**2)
    END

c
    SUBROUTINE VANISH(t,tgt,firer)
c    Vanish: if tgt vanishes treat, otherwise reschedule vanish
    include 'meeting.h'
    integer tgt, firer
    common /terane/ d(2,40), xold(20), yold(20), dist(20), iseg(20)
    rss(x,y)=sqrt(x*x+y*y)

c
    if (trace) print *, '>vanish'
    narmy = army(tgt)
    IF (invisb.eq.1) THEN
        if(speed(tgt).le.0.)print *, 'VANISH: narmy,speed=',narmy,
1        speed(tgt)
        IF (speed(tgt).le.0.) STOP
        call path(tgt,t,motion(tgt),0.0,x,y,vx,vy)
c    Terrain causes intervisibility
        travel = rss(x-xold(tgt), y-yold(tgt))
        IF (travel.gt.dist(tgt)) THEN
c    Tgt is now masked by terrain
            xold(tgt) = x
            yold(tgt) = y
            iseg(tgt) = iseg(tgt)+1
            if (iseg(tgt).gt.40) iseg(tgt)=iseg(tgt)-40
            dist(tgt) = d(narmy,iseg(tgt))
            call vanter(t,tgt,firer)
            dt = dist(tgt)/speed(tgt) + 0.01
            call skedul (t+dt,tgt,'appear',NULL)
        ELSE IF (life(tgt).eq.ALIVE) THEN

```

```

c      Not yet masked by terrain, so reschedule
        dt = (dist(tgt) - travel) / speed(tgt) + 0.01
        call skedul (t+dt,tgt,'vanish',NULL)
      ENDIF
    ELSE
      print *, 'Smoke not played.'
    ENDIF
    if (trace) print *, '<vanish'
  END

c
  SUBROUTINE VANTER(t,tgt,firer)
c 0  Vanter: Treat tgt that vanished behind terrain.
    include 'meeting.h'
    integer tgt, firer
  1   format(f8.2,1x,a4,i3,' vanishes',9x,'(x=',f8.1,' y=',f8.1,')')
c
    if (trace) print *, '>vanter'
    narmy = army(tgt)
    if (keyd(1).ge.2) print 1, t, color(narmy), tgt,
  1   x0(tgt), y0(tgt)
    knceal(tgt) = FD
    nrtgt(tgt) = 0
    ndet(tgt) = 0
c
    Cancel all lines-of-sight and sightings involving tgt
      DO 20 i=1,nblu+nred
        los(tgt,i) = .false.
        los(i,tgt) = .false.
        if (seen(i,tgt)) ndet(i)=ndet(i)-1
        seen(tgt,i) = .false.
        seen(i,tgt) = .false.
        tfire(tgt,i) = 0.0
        tfire(i,tgt) = 0.0
        fot(tgt,i) = .false.
c
      Change by HLReed 1-12-90. Seems to be needed
        busy(tgt) = .false.
  20   CONTINUE
c
    Abort incoming rounds & disengage tanks firing at tgt
      ifirst=1
      if (narmy.eq.1) ifirst = nblu+1
c
      kind = kindrd(3-narmy)
      call newtgt(t,ifirst,tgt)
      call cancel (tgt,'fire ',NULL)
      call cancel (tgt,'select',NULL)
c
    Accelerate tgt that was halting to fire.
      IF (motion(tgt).eq.SLOWNG .and. life(tgt).eq.1) THEN
        call skedul (t,tgt,'accel ',NULL)
        call cancel (tgt,'halt ',NULL)
      ENDIF
    if (trace) print *, '<vanter'
  END

```

```

c      SUBROUTINE XFER(t,i1,j)
c      Xfer detection of firer j to all vehicles on side starting with i1)
      include 'meeting.h'
      i2 = nblu
      if (i1 .ne. 1) i2 = nblu+ nred
      DO 10 i = i1,i2
        IF (life(i) .lt. FKILL .and.
1         ndet(i) .lt. ndets(army(i)) .and.
2         los(i,j) .and.
3         .not.seen(i,j) ) THEN
          seen(i,j) = .true.
          ndet(i) = ndet(i) + 1
          thuman = 0.0 * exp(rolln(0.5))
          call selecs(t,i,thuman)
        ENDIF
10     CONTINUE
      END

c      FUNCTION TDINTP(x1a, x2a, y, x1, x2, ix1a, ix2a )
c      TDINP: Interpolates in a two dimensional matrix.
      integer ix1a, ix2a
      real y(ix1a,ix2a), x1a(ix1a), x2a(ix2a)
      integer j, k
      real y1, y2, y3, y4, t, u

c      j = INDEXX( x1a, ix1a, x1 )
      k = INDEXX( x2a, ix2a, x2 )
      IF(k.eq.0) THEN
        PRINT*, 'TDINTP: p,r,j,k=', x1, x2, j, k
        print *, x1a, x2a, ix1a, ix2a
      ENDIF

c      y1 = y(j,k)
      y2 = y(j+1,k)
      y3 = y(j+1,k+1)
      y4 = y(j,k+1)

c      t = (x1-x1a(j))/(x1a(j+1)-x1a(j))
      u = (x2-x2a(k))/(x2a(k+1)-x2a(k))

c      TDINTP = (1-t)*(1-u)*y1 + t*(1-u)*y2 + t*u*y3 + (1-t)*u*y4
      END

```

APPENDIX B:
CODE FOR POP-UP MODIFICATIONS

INTENTIONALLY LEFT BLANK.

Appendix B

Code for Pop-up Modifications

```
c  MAIN ROUTINE
c  Program changed March 23, 1990 by HLReed for pop up tactics by blue
c  defender. Routines changed are deaths, detect, finish, frdssg,
c  init, init2, and search. New routines are assign, popdown, popup,
c  and stanby. New global variables are ready(NN), npop, tpop, and tmove.
c  include 'common.h'
c
c  open(4, file = '/other/harry/tw/data/pop.dat', status = 'old')
c  rewind 4
c  read(4,*) npop, tpop, tmove
c  close (4)
c  call input
c  call forces
c  END
c
c  SUBROUTINE DEATHS (t)
c  Deaths: Find death toll on each side. A tank is considered
c  .....
c  dead(BLU) = 0
c  dead(RED) = 0
c  Change made March 23, 1990 by HLReed to create popup model
c  Blue player No.1 does not die.
c  DO 20 i=2,(nblu-ndecoy(BLU))
c    dead1 = life(i).ge.IKILL
c    dead2 = knceal(i).eq.FD .and. life(i).ge.FKILL
c    if (dead1 .or. dead2) dead(BLU)=dead(BLU)+1
20  CONTINUE
c  .....
c  END
c
c  SUBROUTINE DETECT (t, firer, tgt)
c  Detect: find if tgt detected and schedule subsequent events.
c  .....
c  t human = 0.0*exp(rolln(0.5))
c  Change march 23,1990 by HLReed to create popup model
c  IF (firer .eq .1) THEN
c    call assign(t)
c  ELSE
c    call selecs(t,firer,thuman)
c  ENDIF
c  IF(xxfer(armyf)) THEN
c  .....
c  END
```

```

c      SUBROUTINE DIS ENG (t, firer, tgt,drop,take)
c      Diseng: attempt to disengage 1 firer from 1 target.
c      .....
c      IF (kind.le.2 .or. kind.eq.5) THEN
c      IF (nrpb(armyf).le.1) THEN
c      Single shot gun system or STAFF fire & forget system.
c      IF (tgt.ne.FLS TGT) THEN
c      IF (fot(firer,tgt)) call cancel (firer,'fire ',tgt)
c      fot(firer,tgt) = .false.
c      Change March 29, 1990 by HLReed to create popup model
c      IF((firer.ge.2).and.(firer.le.nblu)) THEN
c      fot(firer,tgt) = .false.
c      nrtgt(firer) = 0
c      busy(firer) = .true.
c      nrot(firer) = 0
c      call skedul(t + tpop, firer, 'popdwn',NULL)
c      return
c      ENDIF
c      ENDIF
c      hav amo = nrd(firer).lt.nrds(armyf)
c      IF (hav amo) THEN
c      .....
c      END
c
c      SUBROUTINE EVENTS
c      Events: call each event in sequence.
c      .....
c      ELSEIF (iwhat.eq.'reload') THEN
c      call reload (t,iwho)
c      Change march 23,1990 by HLReed to create popup model
c      ELSEIF (iwhat.eq.'popdwn') THEN
c      call popdwn(t,iwho)
c      ELSEIF (iwhat.eq.'popup ') THEN
c      call popup(t,iwho)
c      ELSEIF (iwhat.eq.'stanby') THEN
c      call stanby(t,iwho)
c      ELSEIF (iwhat.eq.'finish') THEN
c      call finish (tm lst)
c      GOTO 99
c      ELSE
c      .....
c      END
c
c      SUBROUTINE FINISH (t)
c      Finish: update statistics at end of a single engagement.
c      .....
c      dalive = 0
c      Change made by H.L. Reed on March 31, 1989 to keep decoys out of
c      win ratios and exchange ratio. See also deaths.f

```

```

c   Change made March 23, 1990 by HLReed to create popup model
      DO 10 i=2,(nblu-ndecoy(BLU))
        k = life(i)
        if (k.ge.5) k=k-1
        nstats(k,BLU) = nstats(k,BLU)+1
        if (life(i).lt.FKILL) balive = balive+1
        brds = brds+nrd(i)
10    CONTINUE
c   .....
      END
c
c   SUBROUTINE FRD SSG (t, firer, tgt, armyf)
c   Frd ssg: Schedule effects after firing single shot gun.
c   .....
      done = nrot(firer).eq.nrpt(armyf)
c   Change March 23, 1990 by HLReed to create popup model
      donpop = nrot(firer).eq.npop
      malive = life(firer) .eq. ALIVE .or. life(firer) .eq. FKILL
      IF((army(firer).eq.BLU) .and. donpop .and. malive) THEN
        fot(firer,tgt) = .false.
        nrtgt(firer) = 0
        busy(firer) = .true.
        nrot(firer) = 0
        call skedul(t + tpop, firer, 'popdown',NULL)
      ELSEIF ((tactc3 .and. done)) THEN
c   .....
      END
c
c   SUBROUTINE INIT
c   Init: Initialize scenario & schedule search at time zero.
c   .....
      DO 20 tgt=1,last
        foes(firer,tgt)= army(firer).ne.army(tgt)
        know(firer,tgt) = 0
        los(firer,tgt) = foes(firer,tgt) .and. invisb.ne.2
c   Change Mar 23, 1990 by HLReed to create popup model
        ready(tgt) = .true.
        if((firer.gt.1).and.(firer.le.nblu)) los(firer,tgt) = .false.
        if((tgt .gt.1).and.(tgt .le.nblu)) los(firer,tgt) = .false.
        mot(firer,tgt) = .false.
        fot(firer,tgt) = .false.
        seen(firer,tgt) = .false.
20    CONTINUE
c   .....
      END
c
c   SUBROUTINE INIT2 (ifirst, last)
c   Init2: initialize each tank on one side.
c   .....
      DO 10 i=ifirst, last

```

```

    army(i) = narmy
    life(i) = ALIVE
    nrd(i) = 0
    nrtgt(i) = 0
    nchan(i) = 0
    nrot(i) = 0
    knceal(i) = jscene
c   Change Mar 23, 1990 by HLReed to create popup model
    if((i.gt.1).and.(i.le.nblu)) knceal(i) = FD
c   Change introduced by HL Reed 8 Mar 89 to allow overwatch tanks to
c   be added to the attacking force. See also subroutines deplo2,
c   input, and cango and common.h.
    if(inwatch(i)) knceal(i) = HD
    ichg(i) = 0
    motion(i) = MAXVL
    if(knceal(i).eq.HD .or. scene.eq.MEETNG) motion(i) = STATNY
c   End of 8 Mar 89 changes.
c   Change Mar 23, 1990 by HLReed to create popup model
    if(knceal(i).eq.FD) motion(i) = STATNY
    nhot(i) = 0
    DO 8 j=1,5
c   .....
    END
c
c   SUBROUTINE SEARCH (t)
c   Search: see if any targets are detected in1 the next second.
c   .....
c   At least one is in detection rg of the other.
    IF (los(i,j)) THEN
c   Line-of-sight exists between them.
c   Treat Southern tank as searcher
c   Change March 23, 1990 by HLReed to create popup model
    IF (rg.lt.rgi .and. .not.seen(i,j) .and.
1      ndet(i).lt.ndeti .and. (j.ne.1)) THEN
        call searc2(t,i,j,iarmy,jcond,dt)
    ELSE
        repeat = .true.
    ENDIF
c   .....
    END
c
c   SUBROUTINE ASSIGN (t)
c   Created March 23, 1990 by HLReed for popup model
    include 'common.h'
    DO 10 i = 2, nblu
    IF(ready(i)) THEN
        ready(i) = .false.
        busy(i) = .true.
        call aprter(t,i,0,HD)
        call skedul(t + tpop, i, 'popup ', NULL)

```

```

    ENDIF
10  CONTINUE
    END

c
    SUBROUTINE POPUP (t,i)
c    Created March 23, 1990 by HLReed for popup model
    include 'common.h'
    DO 10 j = nblu+1, nblu + nred
        seen(i,j) = seen(1,j)
        ndet(i) = ndet(1)
10  CONTINUE
    busy(i) = .false.
    call selects (t,i,0)
    END

c
    SUBROUTINE POPDWN (t,i)
c    Created March 23, 1990 by HLReed for popup model
    include 'common.h'
    call vanter(t,i,0)
    call skedul(t + tmove, i, 'stanby', 0)
    END

c
    SUBROUTINE STANBY (t,i)
c    Created March 23, 1990 by HLReed for popup model
    include 'common.h'
    if(life(i).eq.ALIVE.and.nrd(i).lt.nrds(army(i))) ready(i)=.true.
    END

```

INTENTIONALLY LEFT BLANK.

APPENDIX C:
FORTRAN CODE FOR PREPARATION OF VULNERABILITY DATA

INTENTIONALLY LEFT BLANK.

Appendix C

FORTTRAN code for Preparation of Vulnerability Data

C.1 The program vul.f

```
c    April 25, 1990
c    The program first prompts for the name of the file containing the
c    accuracy data, then for the name of the file containing the
c    probability of kill given a hit data, and finally for the name of
c    the output file.
      dimension sigmax(9,8), sigmay(9,8), ph(8,7), weight(7)
      dimension pkh(8, 2, 11, 4, 7), f(7), disp(8)
      character accfil*32, vulfil*32, outfil*32
c
      print *, 'Accuracy File ='
      read *, accfil
      print *, 'Vulnerability File ='
      read *, vulfil
      print *, 'Output File ='
      read *, outfil
c
c    define the values of the cardioid distribution for 30 deg increments
      weight(1) = .1657
      weight(7) = .001
      do 10 n = 2, 6
        theta = .5236 * (n - 1)
        weight(n) = .16667 + .16477 * cos(theta)
10    continue
c
c    read in accuracy data
      open(4, file = accfil, status = 'old')
      rewind 4
      do 30 kase = 1, 9
        read(4, *) (sigmax(kase, nrange), nrange = 1,8)
        read(4, *) (sigmay(kase, nrange), nrange = 1,8)
30    continue
      close (4)
c
c    read in vulnerability data
      open(4, file = vulfil, status = 'old')
      rewind 4
      do 80 n = 1, 792
        read(4,*,end=81) ir,jex,ndis,ntype,(f(jangle),jangle=1,7)
        if (ir .gt. 0) then
          nrange = ir /500
```

```

        do 90 jangle = 1, 7
            pkh(nrange, jex, ndis, ntype, jangle) = f(jangle)
90        continue
        endif
80    continue
81    close (4)
c
open(4, file = outfil, status = 'new')
do 50 kase = 1, 9
    do 60 jexpos = 1, 2
        do 70 nrange = 1, 8
            sigx = sigmax(kase, nrange) * 0.5 * nrange
            sigy = sigmay(kase, nrange) * 0.5 * nrange
            disp(nrange) = 3.28 * sqrt(sigx * sigy)
            if (disp(nrange) .ge. 11.0) then
                disp(nrange) = 10.999
            else if ( disp(nrange) .le. 1.0) then
                disp(nrange) = 1.001
            endif
            if (jexpos .eq. 1) then
                call phhd(sigx, sigy, ph, nrange)
            else
                call phfe(sigx, sigy, ph, nrange)
            endif
70        continue
            call calkil(ph, pkh, weight, nsig, kase, jexpos, disp)
60        continue
50    continue
end
c
subroutine phhd(sigx, sigy, ph, nrange)
c    probability of hit for hull defilade target
c    half height of turret
parameter (HT = .375)
c    half width of turret
parameter (WT1 = 1.175)
c    half length of turret
parameter (TL1 = 1.475)
dimension ph(8,7)
do 10 j = 1, 7
    theta = .5236 * (j-1)
    wt = WT1 * abs(cos(theta)) + TL1 * abs(sin(theta))
    ph(nrange,j) = (2.*gauss(wt/sigx)-1.)*(2.*gauss(HT/sigy)-1.)
10    continue
end
c
subroutine phfe(sigx, sigy, ph, nrange)
c    probability of nit for fully exposed target
c    half height of turret
parameter (HT = .375)

```

```

c      half width of turret
      parameter (WT1 = 1.175)
c      half length of turret
      parameter (TL1 = 1.475)
c      height of hull
      parameter (HH = 1.5)
c      half width of hull
      parameter (WH1 = 1.775)
c      half length of hull
      parameter (HL1 = 3.375)
      dimension ph(8,7)
      do 10 j = 1, 7
        theta = .5236 * (j-1)
        wt = WT1 * abs(cos(theta)) + TL1 * abs(sin(theta))
        phturr = gauss((.3 + 2.*HT)/sigy) - gauss(.3/sigy)
        phturr = (2.*gauss(wt/sigx) - 1.) * phturr
        phtemp = gauss(.3/sigy) - gauss((.3 - HH)/sigy)
        wh = WH1 * abs(cos(theta)) + HL1 * abs(sin(theta))
        ph(nrange,j) = phtemp * (2. * gauss(wh/sigx) -1.) + phturr
10      continue
      end

c
      function gauss(x)
c      normal distribution function
      parameter (c1 = 1.33027, c2 = 1.821256, c3 = 1.781478)
      parameter (c4 = .3565638, c5 = .3193815)

      y = abs(x)
      g = .398942 * exp(-.5 * x * x)
      if (y .gt. 4.6844 ) then
        g = 1. - g * (1./y - 1./y**3 + 3./y**5)
        if (x .lt. 0) g = 1. - g
        gauss = g
      else
        y = 1./(1. + .2316419*y)
        g = 1. - g * y * (((c1 * y - c2) * y + c3)*y-c4)*y+c5)
        if (x .lt. 0) g = 1. - g
        gauss = g
      endif
      end

c
      subroutine calkil(ph, pkh, weight, nsig, kase, jexpos, disp)
c      calculates kill probabilities amd prints them
      dimension ph(8,7), weight(7), pkh(8, 2, 11, 4, 7), disp(8)
      dimension pk0(8), pk(8,4), pk1(8,4)
      do 10 i = 1, 8
        pk0(i) = 0.0
        do 20 j = 1, 7
          pk0(i) = pk0(i) + weight(j) * ph(i,j)
20        continue

```

```

10    continue
    do 30 k = 1, 4
        do 40 i = 1, 8
            pk(i,k) = 0.0
            do 50 j = 1, 7
                nd = int(dispatch(i))
                d = dispatch(i) - float(nd)
                temp = (1 - d) * pkh(i, jexpos, nd, k, j) + d * pkh(i, jexpos, nd + 1, k, j)
                pk(i, k) = pk(i, k) + weight(j) * ph(i, j) * temp
            50    continue
        40    continue
    30    continue
    do 60 i = 1, 8
        pk1(i, 1) = pk0(i) - pk(i, 4)
        pk1(i, 2) = pk0(i) - pk(i, 1) - pk(i, 2) + pk(i, 3)
        pk1(i, 3) = pk0(i) - pk(i, 2)
        pk1(i, 4) = pk0(i) - pk(i, 3)
    60    continue
1    format(3i2, 9f6.3)
    write(4, 1) kase, jexpos, 0, pk0(1), (pk0(n), n = 1, 8)
    write(4, 1) kase, jexpos, 1, pk1(1, 1), (pk1(n, 1), n = 1, 8)
    write(4, 1) kase, jexpos, 2, pk1(1, 2), (pk1(n, 2), n = 1, 8)
    write(4, 1) kase, jexpos, 3, pk1(1, 3), (pk1(n, 3), n = 1, 8)
    write(4, 1) kase, jexpos, 4, pk1(1, 4), (pk1(n, 4), n = 1, 8)
    end

```

C.2 Input data

The following are the dispersions in mils for the weapon. The eight columns are for ranges of 500 to 4000 meters by 500 meters. The rows are in pairs with the first giving the horizontal dispersion and the second giving the vertical dispersion. In order the cases covered are:

stationary firer and stationary target

stationary firer and moving target

stationary firer and maneuvering target

moving firer and stationary target

moving firer and moving target

moving firer and maneuvering target

maneuvering firer and stationary target

maneuvering firer and moving target

maneuvering firer and maneuvering target

0

0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24
0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24
0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24
0.24	0.24	0.24	0.24	0.24	0.24	0.24	0.24
0.27	0.34	0.42	0.52	0.63	0.73	0.85	0.95
0.27	0.27	0.27	0.27	0.27	0.28	0.28	0.28
0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
0.27	0.27	0.27	0.27	0.27	0.27	0.27	0.27
0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
0.29	0.36	0.44	0.53	0.64	0.74	0.86	0.96
0.29	0.29	0.29	0.29	0.29	0.30	0.30	0.31
0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31
0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.31
0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
0.33	0.38	0.46	0.55	0.65	0.76	0.87	0.97
0.33	0.33	0.33	0.33	0.33	0.34	0.34	0.34

The following is a sample vulnerability file.

The first column gives the range in meters (data are given for 0 to 4000 by 500 meters). The second column gives the exposure where 1 is hull defilade and 2 is fully exposed. The third column gives the dispersion in feet. The fourth column gives the kill type with:

1 = Mobility Kill

2 = Firepower Kill

3 = M or F Kill

4 = K Kill

0

The remaining columns give the Pkh data for azimuths of 0 to 180 degrees by 30 degrees and finally an average over azimuth (the latter is not used).

0,1,1,1,0.006,0.021,0.027,0.045,0.031,0.013,0.006,0.023
0,1,1,2,0.518,0.589,0.715,0.691,0.680,0.671,0.396,0.628
0,1,1,3,0.518,0.589,0.715,0.691,0.680,0.671,0.396,0.628

```

0,1,1,4,0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000
0,1,2,1,0.017,0.030,0.036,0.046,0.036,0.023,0.015,0.031
0,1,2,2,0.343,0.505,0.578,0.569,0.574,0.534,0.268,0.496
0,1,2,3,0.343,0.505,0.578,0.569,0.574,0.534,0.268,0.496
0,1,2,4,0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000
0,1,3,1,0.018,0.029,0.034,0.040,0.033,0.023,0.017,0.029
0,1,3,2,0.296,0.478,0.547,0.534,0.534,0.482,0.233,0.460
0,1,3,3,0.296,0.478,0.547,0.534,0.534,0.482,0.233,0.460
0,1,3,4,0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000
0,1,4,1,0.019,0.028,0.031,0.036,0.030,0.023,0.017,0.028
0,1,4,2,0.277,0.471,0.541,0.526,0.520,0.466,0.219,0.449
0,1,4,3,0.277,0.471,0.541,0.526,0.520,0.466,0.219,0.449
0,1,4,4,0.000,0.000,0.000,0.000,0.000,0.000,0.000,0.000

```

```

.....
4000,2,7,1,0.078,0.273,0.376,0.378,0.410,0.378,0.333,0.277
4000,2,7,2,0.137,0.259,0.344,0.370,0.378,0.364,0.336,0.278
4000,2,7,3,0.163,0.377,0.494,0.505,0.528,0.493,0.396,0.385
4000,2,7,4,0.025,0.113,0.144,0.135,0.159,0.149,0.189,0.105
4000,2,8,1,0.079,0.266,0.369,0.376,0.404,0.369,0.328,0.273
4000,2,8,2,0.135,0.251,0.332,0.361,0.366,0.352,0.329,0.270
4000,2,8,3,0.162,0.371,0.488,0.505,0.522,0.483,0.390,0.381
4000,2,8,4,0.025,0.107,0.135,0.129,0.151,0.142,0.184,0.100
4000,2,9,1,0.079,0.262,0.363,0.374,0.399,0.363,0.324,0.270
4000,2,9,2,0.134,0.246,0.324,0.355,0.357,0.344,0.324,0.265
4000,2,9,3,0.162,0.367,0.485,0.506,0.518,0.476,0.385,0.379
4000,2,9,4,0.024,0.103,0.129,0.124,0.145,0.137,0.181,0.096
4000,2,10,1,0.079,0.259,0.359,0.373,0.395,0.358,0.321,0.268
4000,2,10,2,0.133,0.242,0.319,0.352,0.351,0.338,0.321,0.261
4000,2,10,3,0.162,0.364,0.482,0.507,0.516,0.471,0.382,0.378
4000,2,10,4,0.024,0.100,0.125,0.121,0.141,0.133,0.179,0.093
4000,2,11,1,0.080,0.244,0.337,0.358,0.373,0.335,0.309,0.255
4000,2,11,2,0.129,0.227,0.303,0.351,0.332,0.312,0.306,0.251
4000,2,11,3,0.160,0.352,0.477,0.523,0.508,0.449,0.369,0.375
4000,2,11,4,0.023,0.087,0.104,0.104,0.120,0.117,0.170,0.080

```

C.3 Typical Output

The following is a typical output from the vulnerability preprocessing program. The first column gives the case where case = 1 is for stationary firer and stationary target and case 9 is for maneuvering firer and maneuvering target. The second column is the exposure with 1 = hull defilade and 2 = fully exposed. the third column relates to the argument of $p(n)$ as discussed in the text. There is displacement of values: 0 relates to $p(1)$, 1 to $p(2)$, etc. The reader is reminded that these are not simple kill probabilities but are numbers that divide the unit interval to represent statistically independent events that relate to probabilities of various combinations of kills.

1	1	0	0.998	0.998	0.882	0.702	0.564	0.462	0.385	0.324	0.274
1	1	1	0.998	0.998	0.882	0.702	0.564	0.462	0.385	0.324	0.274
1	1	2	0.974	0.974	0.861	0.684	0.548	0.448	0.373	0.314	0.266
1	1	3	0.377	0.377	0.346	0.298	0.271	0.243	0.209	0.183	0.160
1	1	4	0.377	0.377	0.346	0.298	0.271	0.243	0.209	0.183	0.160
1	2	0	1.000	1.000	1.000	0.998	0.979	0.934	0.870	0.799	0.728
1	2	1	0.607	0.607	0.608	0.630	0.670	0.687	0.665	0.632	0.592
1	2	2	0.380	0.380	0.380	0.417	0.493	0.547	0.547	0.538	0.515
1	2	3	0.315	0.315	0.316	0.346	0.407	0.451	0.454	0.451	0.435
1	2	4	0.301	0.301	0.301	0.328	0.383	0.421	0.417	0.404	0.385
2	1	0	0.998	0.998	0.882	0.702	0.564	0.462	0.385	0.324	0.274
2	1	1	0.998	0.998	0.882	0.702	0.564	0.462	0.385	0.324	0.274
2	1	2	0.974	0.974	0.861	0.684	0.548	0.448	0.373	0.314	0.266
2	1	3	0.377	0.377	0.346	0.298	0.271	0.243	0.209	0.183	0.160
2	1	4	0.377	0.377	0.346	0.298	0.271	0.243	0.209	0.183	0.160
2	2	0	1.000	1.000	1.000	0.998	0.979	0.934	0.870	0.799	0.728
2	2	1	0.607	0.607	0.608	0.630	0.670	0.687	0.665	0.632	0.592
2	2	2	0.380	0.380	0.380	0.417	0.493	0.547	0.547	0.538	0.515
2	2	3	0.315	0.315	0.316	0.346	0.407	0.451	0.454	0.451	0.435
2	2	4	0.301	0.301	0.301	0.328	0.383	0.421	0.417	0.404	0.385
3	1	0	0.995	0.995	0.835	0.635	0.449	0.295	0.188	0.125	0.088
3	1	1	0.995	0.995	0.835	0.635	0.449	0.295	0.188	0.125	0.088
3	1	2	0.970	0.970	0.815	0.616	0.435	0.286	0.183	0.122	0.086
3	1	3	0.375	0.375	0.328	0.302	0.240	0.163	0.106	0.071	0.050
3	1	4	0.375	0.375	0.328	0.302	0.240	0.163	0.106	0.071	0.050
3	2	0	1.000	1.000	1.000	0.989	0.917	0.779	0.609	0.460	0.349
3	2	1	0.607	0.607	0.608	0.685	0.702	0.632	0.513	0.397	0.306
3	2	2	0.380	0.380	0.380	0.511	0.581	0.548	0.458	0.360	0.281
3	2	3	0.315	0.315	0.315	0.421	0.480	0.458	0.387	0.308	0.242
3	2	4	0.301	0.301	0.301	0.394	0.437	0.401	0.329	0.257	0.199
4	1	0	0.996	0.996	0.851	0.663	0.526	0.426	0.350	0.290	0.243
4	1	1	0.996	0.996	0.851	0.663	0.526	0.426	0.350	0.290	0.243
4	1	2	0.972	0.972	0.830	0.646	0.510	0.413	0.339	0.282	0.236
4	1	3	0.376	0.376	0.334	0.290	0.262	0.227	0.192	0.166	0.142
4	1	4	0.376	0.376	0.334	0.290	0.262	0.227	0.192	0.166	0.142
4	2	0	1.000	1.000	1.000	0.995	0.966	0.907	0.832	0.754	0.679
4	2	1	0.607	0.607	0.608	0.644	0.682	0.681	0.648	0.609	0.559
4	2	2	0.380	0.380	0.380	0.442	0.519	0.552	0.543	0.527	0.491
4	2	3	0.315	0.315	0.316	0.365	0.428	0.456	0.452	0.442	0.416
4	2	4	0.301	0.301	0.301	0.345	0.401	0.422	0.411	0.393	0.365
5	1	0	0.996	0.996	0.851	0.663	0.526	0.426	0.350	0.290	0.243
5	1	1	0.996	0.996	0.851	0.663	0.526	0.426	0.350	0.290	0.243
5	1	2	0.972	0.972	0.830	0.646	0.510	0.413	0.339	0.282	0.236
5	1	3	0.376	0.376	0.334	0.290	0.262	0.227	0.192	0.166	0.142
5	1	4	0.376	0.376	0.334	0.290	0.262	0.227	0.192	0.166	0.142
5	2	0	1.000	1.000	1.000	0.995	0.966	0.907	0.832	0.754	0.679
5	2	1	0.607	0.607	0.608	0.644	0.682	0.681	0.648	0.609	0.559
5	2	2	0.380	0.380	0.380	0.442	0.519	0.552	0.543	0.527	0.491

5	2	3	0.315	0.315	0.316	0.365	0.428	0.456	0.452	0.442	0.416
5	2	4	0.301	0.301	0.301	0.345	0.401	0.422	0.411	0.393	0.365
6	1	0	0.990	0.990	0.804	0.599	0.419	0.273	0.174	0.116	0.079
6	1	1	0.990	0.990	0.804	0.599	0.419	0.273	0.174	0.116	0.079
6	1	2	0.966	0.966	0.784	0.581	0.406	0.265	0.170	0.113	0.077
6	1	3	0.374	0.374	0.321	0.291	0.226	0.152	0.098	0.066	0.045
6	1	4	0.374	0.374	0.321	0.291	0.226	0.152	0.098	0.066	0.045
6	2	0	1.000	1.000	1.000	0.983	0.899	0.752	0.582	0.435	0.321
6	2	1	0.607	0.607	0.616	0.694	0.695	0.614	0.492	0.377	0.282
6	2	2	0.380	0.380	0.392	0.529	0.579	0.535	0.440	0.343	0.260
6	2	3	0.315	0.315	0.325	0.435	0.480	0.448	0.373	0.294	0.224
6	2	4	0.301	0.301	0.310	0.406	0.434	0.391	0.316	0.244	0.184
7	1	0	0.988	0.988	0.789	0.594	0.461	0.366	0.295	0.240	0.198
7	1	1	0.988	0.988	0.789	0.594	0.461	0.366	0.295	0.240	0.198
7	1	2	0.963	0.963	0.770	0.577	0.447	0.355	0.286	0.233	0.192
7	1	3	0.373	0.373	0.309	0.273	0.242	0.198	0.165	0.138	0.116
7	1	4	0.373	0.373	0.309	0.273	0.242	0.198	0.165	0.138	0.116
7	2	0	1.000	1.000	1.000	0.986	0.933	0.851	0.760	0.674	0.597
7	2	1	0.607	0.607	0.608	0.663	0.690	0.656	0.611	0.554	0.501
7	2	2	0.380	0.380	0.380	0.478	0.552	0.545	0.525	0.486	0.447
7	2	3	0.315	0.315	0.315	0.394	0.454	0.452	0.439	0.410	0.381
7	2	4	0.301	0.301	0.301	0.371	0.423	0.412	0.392	0.359	0.329
8	1	0	0.988	0.988	0.789	0.594	0.461	0.366	0.295	0.240	0.198
8	1	1	0.988	0.988	0.789	0.594	0.461	0.366	0.295	0.240	0.198
8	1	2	0.963	0.963	0.770	0.577	0.447	0.355	0.286	0.233	0.192
8	1	3	0.373	0.373	0.309	0.273	0.242	0.198	0.165	0.138	0.116
8	1	4	0.373	0.373	0.309	0.273	0.242	0.198	0.165	0.138	0.116
8	2	0	1.000	1.000	1.000	0.986	0.933	0.851	0.760	0.674	0.597
8	2	1	0.607	0.607	0.608	0.663	0.690	0.656	0.611	0.554	0.501
8	2	2	0.380	0.380	0.380	0.478	0.552	0.545	0.525	0.486	0.447
8	2	3	0.315	0.315	0.315	0.394	0.454	0.452	0.439	0.410	0.381
8	2	4	0.301	0.301	0.301	0.371	0.423	0.412	0.392	0.359	0.329
9	1	0	0.977	0.977	0.744	0.537	0.368	0.240	0.151	0.102	0.071
9	1	1	0.977	0.977	0.744	0.537	0.368	0.240	0.151	0.102	0.071
9	1	2	0.953	0.953	0.725	0.520	0.357	0.233	0.147	0.099	0.070
9	1	3	0.369	0.369	0.306	0.270	0.201	0.134	0.085	0.058	0.040
9	1	4	0.369	0.369	0.306	0.270	0.201	0.134	0.085	0.058	0.040
9	2	0	1.000	1.000	0.999	0.968	0.858	0.702	0.530	0.395	0.296
9	2	1	0.607	0.607	0.628	0.702	0.675	0.580	0.452	0.344	0.261
9	2	2	0.380	0.380	0.413	0.552	0.572	0.510	0.408	0.314	0.241
9	2	3	0.315	0.315	0.342	0.453	0.475	0.428	0.346	0.269	0.208
9	2	4	0.301	0.301	0.325	0.422	0.426	0.370	0.291	0.223	0.170

**APPENDIX D:
MISCELLANEOUS CODE FOR RUNNING**

INTENTIONALLY LEFT BLANK.

Appendix D

Miscellaneous Code for Running

D.1 Code to Use with TWMEET

The program is evoked with a command line such as:

```
runtwmeet f1 f2 f3 f4 f5 f6 f7 f8
```

where

f1 is the game file

f2 is the miscellaneous file for blue

f3 is the vulnerability file for blue (to kill red)

f4 is the file that describes blue's priority scheme

f5 is the miscellaneous file for red

f6 is the vulnerability file for red

f7 is the priority file for red

f8 is the output file

0

The program runtwmeet is contained in the shell file

```
cp $2 blue.misc
cp $3 blue.vul
cp $4 blue.pri
cp $5 red.misc
cp $6 red.vul
cp $7 red.pri
echo Defender: $2;" $3;" $4> $8
echo Attacker: $5;" $6;" $7>> $8
twmeet < $1 >> $8
```

0

Where blue.misc, blue.vul, blue.pri, red.misc, red.vul, red.pri are reserved as files used by the program for temporary storage. twmeet is the code that is compiled from twmeet.f. The "echoes" are used to put the names of the input files into the output data.

The game file is the same as the game file for TANKWARS except that the first two lines replace the first three lines of the game file for TANKWARS, the misc and vul files are predefined, and the predefined priority files are added at the end (the user leaves these files alone except perhaps to make sure that his path definitions are OK). A typical file follows:

```
0,3
9,0
0 0 0 0 0    Lev,echo, ,trace,sked
0 0 0 0 0    Indices of print flags to turn on
1000,4000,1000,500,    Open range loop control
1000,1,2,1,1111111,    #reps,#waves,#dist,meth,random seed
999.,        max time
blue.misc
1blue.vul
red.misc
1red.vul
1,3,    Terrain/Smoke,#Smoke data lines
blue.pri
red.pri
```

0

Note that this file assigns 3 defending tanks to blue and 9 attacking tanks to red.

The miscellaneous files which define the characteristics of the vehicles on each side are very similar to those used in TANKWARS.

All the entries for TANKWARS must be filled in - there could be some clean up such as for decoys - but the program TWMEET still looks for the data even if it doesn't use it. This is a consequence of not wanting to make continual changes in input files as the program developed. TWMEET does not use the data on line 2. There is an added entry at the end of line 9; it is the median time for a vehicle to pinpoint a firing target. There is an added entry on line 18; it determines whether or not a target is transferred from the vehicle that detects it to the other vehicles on the same side. The following is a typical miscellaneous file:

```
0.75,1.175,1.475,1.475,1.5,1.775,3.375,3.375,    1 Turret & Hull dimensions
0.,0.,0.,0.,0.,0.,0.,    2 Prob of sensing a miss
```

```

1.00, 1.00 0.59, 0.24, 0.10, 0.04, 0.01, 0.01 HD p infinity
1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00 FE p infinity
1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00 Mov p infinity
21.73, 46.32, 83.62, 203.6, 501.7, 1228., 2814., 5000. HD t bar
2.06, 4.32, 6.84, 9.66, 12.89, 16.63, 21.02, 26.26 FE t bar
2.06, 4.32, 6.84, 9.66, 12.89, 16.63, 21.02, 26.26 Mov t bar
1500.,0.,0.60.,.24.,99,150.,4.0 9 Recnz, pfalse(HD|FE), tlook, pinp, reliab, trelod
40,1,1,1,1,0,1, 10 See description
0.3,.6,.91,1.22,1.55,1.87,2.1,2.4, 11 Time of flight
3.7, 3.7, 3.7, 3.7, 3.7, 3.7, 3.7, 3.7 12 T-first
2.25, 10., 0.0 13 Firing cycle parameters
0.,0.,0.,0.,0.,0.,0.,0., 14 Tfixed
1.,2.,5.,180.,60., 15 Accel,decel,speed,angle,thide
0,3,25, 16 Halt-to-fire, disengage values
0,0, 17 #of decoys, #of flashing decoys
F,F
I
9

```

A priority file has 192 lines. The first 6 columns are the n1, ... , n6 discussed in the text and the last column is the priority for that case with lower numbers representing higher priorities. A typical priority file looks like:

```

1 1 1 1 1 1 0
1 1 1 1 1 2 1
1 1 1 1 1 3 0
1 1 1 1 1 4 3
2 1 1 1 1 1 0
2 1 1 1 1 2 1
2 1 1 1 1 3 0
2 1 1 1 1 4 3
.....
2 1 2 2 3 3 0
2 1 2 2 3 4 63
2 2 2 2 3 1 0
2 2 2 2 3 2 61
2 2 2 2 3 3 0
2 2 2 2 3 4 63

```

0
Finally, a typical output file looks like:

```

Defender: t24.t1.06; v1.var1f; pri.1
Attacker: t24.t1.06; v1.var1f; pri.1
1000. 2.993 2.825 1 99 0.94 1.935 1.274
2000. 2.847 3.858 8 92 1.36 4.329 2.201
3000. 2.848 3.611 9 91 1.27 6.384 3.677
4000. 2.825 3.468 9 91 1.23 8.895 5.704

```

0

The first lines give the miscellaneous, vulnerability, and priority files for each side. The columns for the rest of the data give the opening range in meters, the number of defenders (blue) killed per engagement, the number of attackers killed, the percent of defender wins, the percent of attacker wins, the exchange ratio (attackers killed / defenders killed), the average number of rounds fired by each defender, and the average number of rounds fired by each attacker.

D.2 Code to Use with Popup Model

The program for pop-up resembles TANKWARS more closely than does TWMEET. The command line used is

```
runtwpop fgame "3 5. 10." fmiscb fvu1b fmiscr fvulr fout
```

where there are no priority files and the numbers in quotation marks are the values of npop, tpop, and tmove.

The program runtwpop is:

```

echo $2 > $7
echo $2 > pop.dat
cp $3 blue.misc
cp $4 x/blue.vul
cp $5 x/red.misc
cp $6 x/red.vul
echo Defender: $3;" $4 >> $7
echo Attacker: $5;" $6 >> $7
twpop1 < $1 >> $7

```

No of Copies	Organization
1	Office of the Secretary of Defense OUSD(A) Director, Live Fire Testing ATTN: James F. O'Bryon Washington, DC 20301-3110
2	Administrator Defense Technical Info Center ATTN: DTIC-DDA Cameron Station Alexandria, VA 22304-6145
1	HQDA (SARD-TR) WASH DC 20310-0001
1	Commander US Army Materiel Command ATTN: AMCDRA-ST 5001 Eisenhower Avenue Alexandria, VA 22333-0001
1	Commander US Army Laboratory Command ATTN: AMSLC-DL Adelphi, MD 20783-1145
2	Commander US Army, ARDEC ATTN: SMCAR-IMI-I Picatinny Arsenal, NJ 07806-5000
2	Commander US Army, ARDEC ATTN: SMCAR-TDC Picatinny Arsenal, NJ 07806-5000
1	Director Benet Weapons Laboratory US Army, ARDEC ATTN: SMCAR-CCB-TL Watervliet, NY 12189-4050
1	Commander US Army Armament, Munitions and Chemical Command ATTN: SMCAR-ESP-L Rock Island, IL 61299-5000
1	Commander US Army Aviation Systems Command ATTN: AMSAV-DACL 4300 Goodfellow Blvd. St. Louis, MO 63120-1798

No of Copies	Organization
1	Director US Army Aviation Research and Technology Activity ATTN: SAVRT-R (Library) M/S 219-3 Ames Research Center Moffett Field, CA 94035-1000
1	Commander US Army Missile Command ATTN: AMSMI-RD-CS-R (DOC) Redstone Arsenal, AL 35898-5010
1	Commander US Army Tank-Automotive Command ATTN: AMSTA-TSL (Technical Library) Warren, MI 48397-5000
1	Director US Army TRADOC Analysis Command ATTN: ATAA-SL White Sands Missile Range, NM 88002-5502
(Class. only) 1	Commandant US Army Infantry School ATTN: ATSH-CD (Security Mgr.) Fort Benning, GA 31905-5660
(Unclass. only) 1	Commandant US Army Infantry School ATTN: ATSH-CD-CSO-OR Fort Benning, GA 31905-5660
1	Air Force Armament Laboratory ATTN: AFATL/DLODL Eglin AFB, FL 32542-5000 <u>Aberdeen Proving Ground</u>
2	Dir, USAMSAA ATTN: AMXSY-D AMXSY-MP, H. Cohen
1	Cdr, USATECOM ATTN: AMSTE-TD
3	Cdr, CRDEC, AMCCOM ATTN: SMCCR-RSP-A SMCCR-MU SMCCR-MSI
1	Dir, VLAMO ATTN: AMSLC-VL-D

<u>No of Copies</u>	<u>Organization</u>
2	Commander US Army Laboratory Command ATTN: AMSLC-TP-PB, I. Bartky, F. Ostovic Adelphi, MD 20783-1145
7	Commander US Army, ARDEC ATTN: SMCAR-TD SMCAR-TDT SMCAR-SCF, V. Galgano E. Del Coco M. Barbarisi SMCAR-CCS SMCAR-CCL-FW, H. Kahn Picatinny Arsenal, NJ 07806-5000
1	Commander US Army Watervliet Arsenal ATTN: SMCWV-QAR, Building 44 W. Jarrett Watervliet, NY 12189
6	Commander US Army Tank Automotive Command ATTN: AMSTA-RY, R. Beck AMCPM-BFVS AMCPM-ABMS AMCPM-ABMS-SW AMCPM-ABMS-SI, G. Vander Waerden Warren, MI 48090
2	Commandant US Army Infantry School ATTN: ATSH-TSM-FV ATSH-CD-MLS-M Fort Benning, GA 31095

<u>No. of Copies</u>	<u>Organization</u>
3	Commander US Army Armor Center ATTN: ATSB-CD ATZK-CD-MS, Mr Falkovich ATZK-AE-PD, Mr. Wells Fort Knox, KY 40121
1	Commander TRAC RPD ATTN: ATRC-DCS Fort Monroe, VA 23651
1	Director Development Center ATTN: MCDEC/DO92, Firepower Division Quantico, VA 22134
1	Director Benet Weapons Laboratory US Army, ARDEC ATTN: SMCAR-CCB Watervliet, NY 12189
2	PM-TMAS ATTN: DRCPM-TMA-PA, K. Rubin DRCPM-TMA, F. Steinberg Picatinny Arsenal, NJ 07806
1	OUSD(A) ATTN: Executive Director, DSB Room 3D1020, Pentagon Washington, DC 20301
1	OUSD(A) ATTN: DUSD(R&AT) Room 3E114, Pentagon Washington, DC 20301
1	OUSD(A) ATTN: DUSD(TWP) Room 3E1044, Pentagon Washington, DC 20301

No. of Copies	Organization
2	ADUSD ATTN: TWP/LW, Room 3D1049/Viilu TWP/AW, Room 3E1049/Loder Pentagon Washington, DC 20301
1	Assistant Secretary of Defense of C ³¹ Room 3E172, Pentagon Washington, DC 20301
1	OUSD(A) ATTN: Director, Program Integration Room 3E1034/Christie, Pentagon Washington, DC 20301
1	Director, PA&E ATTN: Director, LFD Room 2B256, Pentagon Washington, DC 20301
1	Director DARPA 1400 Wilson Blvd. Rosslyn, VA 22209
1	HQDA (SAUS-OR, Mr. Hollis) WASH DC 20310-0001
1	HQDA (SARD-ZD) WASH DC 20310-0001
1	Commander USAMC ATTN: AMCCG 5001 Eisenhower Avenue Alexandria, VA 22333-0001
1	Commander AVSCOM ATTN: AMCPM-AAH 4300 Goodfellow Blvd. St. Louis, MO 63120-1798
1	Commander CECOM ATTN: AMSEL-RD-EW-D Fort Monmouth, NJ 07703-5303

No. of Copies	Organization
1	Commander CNVEO ATTN: AMSEL-RD-NV-T Fort Belvoir, VA 22060-5166
1	Commander ASL/LABCOM ATTN: SLCAS-D White Sands Missile Range, NM 88002-5501
1	Commander VAL/LABCOM ATTN: SLCVA-D White Sands Missile Range, NM 88002-5513
2	Commander US Army Missile Command ATTN: AMCPEO-FS AMSMI-RD-AS Redstone Arsenal, AL 35898-5001
1	Director Survivability Management Office ATTN: SLCSM-GS 2800 Powder Mill Road Adelphi, MD 20783-1145
1	Commander USA FSTC ATTN: William Rich 220 Seventh St., NE Charlottesville, VA 22901-5396
1	Commander TRAC RPD ATTN: ATCG Fort Monroe, VA 23651-5000
1	Commander USA Air Defense Artillery School ATTN: ATSA-CD Fort Bliss, TX 79916

<u>No. of Copies</u>	<u>Organization</u>
1	Commander USA Armor Center and Fort Knox ATTN: ATSB-CD Fort Knox, KY 40121-5000
1	Commander USA Combined Arms Combat Development Activity ATTN: ATZL-CA Fort Leavenworth, KS 66027-5300
1	Commander USAAVNC and Fort Rucker ATTN: ATZQ-TDS Fort Rucker, AL 36362-5163
1	Commandant USA Infantry School ATTN: ATSH-CD-MLS Fort Benning, GA 31905-5000
1	Commander USA Intelligence Center and School ATTN: ATSI-CD-TE Fort Haachuca, AZ 85613-7000
1	Director TRAC-WSMR White Sands Missile Range, NM 88002-5502
1	Assistant Secretary of the Navy Research, Engineering and Systems Room 4E736, Pentagon Washington, DC 20350
1	Commandant USMC ATTN: Code RD Washington, DC 20380-0001
1	Deputy Commanding General MCRDAC ATTN: Code D032 Quantico, VA 22134-5080
1	Director USMC OTEA Quantico, VA 22134

<u>No. of Copies</u>	<u>Organization</u>
1	Director DIA ATTN: RTS-2, Col MacNicoll Pentagon Washington, DC 20301-6111
2	Director CIA ATTN: Mr. John Ewen/Mr. Robert Gomez P.O. Box 1925 Washington, DC 20505
1	Joint Electronic Warfare Center ATTN: MAJ Harry Ladewig San Antonio, TX 78243-5000
3	Los Alamos National Laboratory ATTN: CDT (J. Immele) ATAC (F. Day, P. Howe) Los Alamos, NM 87545
1	Lawrence Livermore National Laboratory ATTN: Dr. Milton Finger Livermore, CA 94550
1	Geometric Solutions ATTN: Mr. Harry L. Reed, Jr. 100 Custus Street Aberdeen, MD 21001
	<u>Aberdeen Proving Ground</u>
1	Dir, USAMSAA ATTN: Mr. T. Ruth
1	Dir, USAHEL ATTN: J. Waugh
1	Cdr, USACSTA ATTN: STCS-CC-P, P. McCall
1	Commander Project Manager Smoke/Obscurants ATTN: AMCPEO-CNS-CT

USER EVALUATION SHEET/CHANGE OF ADDRESS

This Laboratory undertakes a continuing effort to improve the quality of the reports it publishes. Your comments/answers to the items/questions below will aid us in our efforts.

1. BRL Report Number BRL-CR-641 Date of Report SEPTEMBER 1990
2. Date Report Received _____
3. Does this report satisfy a need? (Comment on purpose, related project, or other area of interest for which the report will be used.) _____

4. Specifically, how is the report being used? (Information source, design data, procedure, source of ideas, etc.) _____

5. Has the information in this report led to any quantitative savings as far as man-hours or dollars saved, operating costs avoided, or efficiencies achieved, etc? If so, please elaborate. _____

6. General Comments. What do you think should be changed to improve future reports? (Indicate changes to organization, technical content, format, etc.) _____

CURRENT ADDRESS

Name

Organization

Address

City, State, Zip Code

7. If indicating a Change of Address or Address Correction, please provide the New or Correct Address in Block 6 above and the Old or Incorrect address below.

OLD ADDRESS

Name

Organization

Address

City, State, Zip Code

(Remove this sheet, fold as indicated, staple or tape closed, and mail.)

-----FOLD HERE-----

DEPARTMENT OF THE ARMY

Director

U.S. Army Ballistic Research Laboratory

ATTN: SLCBR-DD-T

Aberdeen Proving Ground, MD 21005-5066

OFFICIAL BUSINESS



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT No 0001, APG, MD

POSTAGE WILL BE PAID BY ADDRESSEE

Director

U.S. Army Ballistic Research Laboratory

ATTN: SLCBR-DD-T

Aberdeen Proving Ground, MD 21005-9989



-----FOLD HERE-----