02921D

Program Description I

Program Title LEM LANDER-- An Orbital Mechanics Game

Contributor's Name Alan Stein

Address 372 Shotwell St.

City S.F. **State** CA. **Zip Code** 94110

Program Description, Equations, Variables This program simulates a Lunar Excursion Module in orbit 100 km above the surface of the moon. The object is to execute a soft-landing (velocity less than 5m/sec, at an angle not more than 5° from vertical) given a limited supply of fuel. On each move, you have the option of either free-falling for a specified period of time, or applying a specified thrust during a specified time period. Time is measured in seconds, thrust as a delta-vee in m/sec, in a given direction from 0° to ±180°. You are not allowed to apply a thrust of greaterthan 7 Gees(69 m/sec/sec of time period). If you run out of fuel, your thrust will be reduced to the fuel supply on hand. Thereafter, any thrust value you provide will be automatically changed to zero. When you pass zero altitude (i.e. land or crash), the program will calculate your velocity at impact and display that automatically.

In general, an orbit will be defined by the analytic equations which characterize an orbit in terms of its energy, E, and its angular momentum, l . These equations are time-independant -- this means that it is possible to specify very long time periods (such as 2000 sec) and get an accurate computation of your new position. (Most such games use vector-sum equations which become exceedingly innacurate for long jumps.). Because the orbital equations are time-independant, the program has to convert the desired "delta-t" into a variable the equations can work with. This conversion process is not completely accurate, but the only error it introduces is that the actual duration of the jump may be slightly different from the one you specify. No postional error is introduced-- you will still be on exactly the correct orbit-- but you will find yourself at a slghtly different point along along that orbit. For example, a 2000 second jump along the initial orbit will take you almost a third of the way around the moon, but the conversion approximation will be about 10% shorter than an actual 2000 sec. jump.

The equations do not work well for degenerate orbits(such as the final vertical descent!), so the program automatically switches to a set of vector-sum equations for the final descent. These equations are highly accurate under such conditions, so large jumps are still possible.

Program Description I

THE FOLLOWING VARIABLES ARE USED: Position: radius(R_i) and radial angle (θ_i). Velocity: speed (V) and direction (θ_v), or horizontal and radial components (V_θ, V_r). Energy: (E). Angular momentum: (l). Eccentricity of the conic section corresponding to the orbit: (e). Orientation of the orbit conic: (θ'). Gravitational constant: (Gm). The orbital conditions are completely specified by the position, velocity, and gravitational constant for the moon. All other variables may be calculated from these. Initially, Gm = 4.89663*10**12

In addition, the fuel supply should be set to 2000, R_{surf} = 1739000, and register(I) = 360.

R_i=1839000 m.
θ_i= 0 θ_v= 0
V=1631.765625 m/sec.

Once these variables are initialized, they may be stored on a spare card for future use. By changing the initial values, different orbits or planets may be experimented with.

Equations: Given a thrust = $d\vec{V}$, the new initial velocity: $\vec{V} = \vec{V}_{old} + d\vec{V}$.

then: $E = \frac{1}{2}V^2 - Gm/R_i$

$l = R_i V_\theta$

$R_o = l^2/Gm$

$e = \sqrt{(1 + 2ER_o/Gm)}$

$\theta' = \theta_i + \text{acos}((R_o/R_i - 1)/e)$

Given a delta-tee = t, we find the new θ_i as follows:

$\vec{R}' = \vec{R} + \vec{V}t - \frac{1}{2}Gmt^2/R_i^2\hat{r}$

$\equiv (R', d\theta)$

Then $\theta_i \approx \theta_{i\,old} + d\theta$

Given this, we compute $R_{inew} = R_o/(1+e*\cos(\theta_i - \theta'))$; $V_{new} = \sqrt{2(E + Gm/R_{new})}$ $\theta_{v\,new} = \text{acos}(l/V_{new}*R_{new})$.

If $V_\theta < 9$ m/sec., a vector sum formula is used : $\vec{V}_{new} = \vec{V} - \frac{\hat{r}Gm}{R^2}*t + d\vec{V}$

$\vec{R}_{new} = \vec{R} + (\vec{V}_{new} - \frac{1}{2}(\frac{\hat{r}Gm}{R^2}*t + d\vec{V}))*t$. If the altitude (a) at the end of a period is negative, then V_r is recomputed back to zero altitude by the approximation: $V_{rzero} = (V_r^2 + 3a)$.

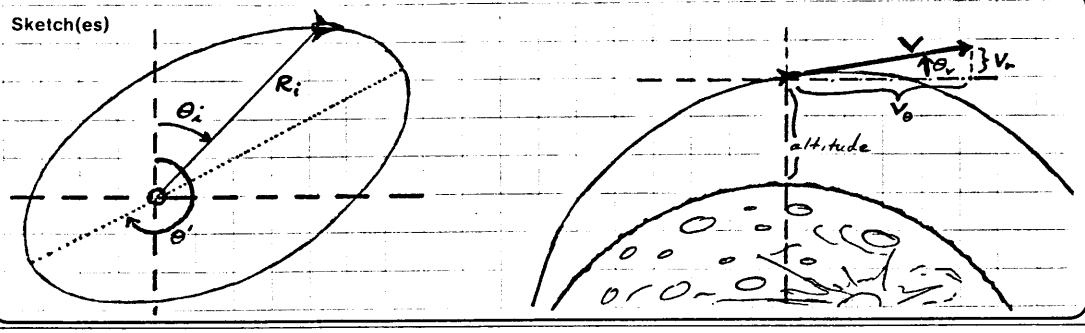
Operating Limits and Warnings Thrust must be less than 7 Gees (69 m/sec of delta-vee for each second of the thrusting period). The program will reject larger thrusts.

Landing velocities greater than 5 m/sec and more than 5° from true vertical (90°) are considered crashes.

At the beginning of a game, Flag 1 must be set. This is done automatically upon loading the program card. If you start a new game, be sure to Set Flag 1 manually at the same time, or reload the program card.

02921D Program Description II

Sketch(es)



Sample Problem(s) 1) YOU ARE IN A 100 Km. LUNAR ORBIT. YOU MAKE A RETROFIRE BURN OF 1000 m/SEC. FIND YOUR POSITION AT 50 SECOND INTERVALS.

NOTE: To initialize standard lunar orbit (100 km-circular), clear primary registers and load the following variables:

R_0 = 2000 (fuel supply)
R_1 = 1839000 (R_i in meters)
R_3 = 4.89663 * 10**12 (Gm)
R_5 = 1631.765625 (V_e in meters/sec)
R_B = 1739000 (lunar radius in meters)
R_I = 360 (°'s in full circle)

You may then store the primary registers on one side of a data card for future use.

Solution(s) LOAD PROGRAM CARD; LOAD DATA CARD; 1000, ENT, 180, ENT, 50, B; (displayed:) altitude=98,460; V=635 m/sec; θ_v=-5.56°; fuel supply = 1000. (stack may be rolled up repeatedly to review the data). 50; A ; alt=93,835; V=646; θ_v=-11. 50; A ; alt= 86,112; V=663; θ_v=-16.25. 50; A ; alt=75266 ; V=687 θ_v = -21.2 . 50; A ; alt = 61,262; V= 716; θ_v= -25.8. 50; A ; alt.=44,057 ; V=752 ; θ_v=-30. 50; A ; alt.=23,593 ; V =794; θ_v= -33.8 . 50; A ; alt.= 0.000 (IMPACT); V_{impact}= 839.883 m/sec. θ_v= -37.293° ; fuel supply = 1000 m/sec. TO PLAY ANOTHER GAME, RELOAD DATA CARD, AND EITHER SET FLAG 1 OR RELOAD PROGRAM CARD.

Reference(s) much of the programming and equation set in this game are derived from 00667D-- Delta-V - Orbit Simulator by Harold T. Coderre.

User Instructions

STEP	INSTRUCTIONS	INPUT DATA/UNITS	KEYS	OUTPUT DATA/UNITS
1	LOAD BOTH SIDES OF PROGRAM CARD	---	--- ---	---
2	INITIALIZE REGISTERS	see		-
	OR	sample		-
2	LOAD DATA CARD	problem		-
3	ENTER THRUST MAGNITUDE (m/sec of delta vee), THRUST ANGLE (degrees: 0 would mean a thrust straight ahead, 90 would be a thrust directly up, 180 is a retrofire) and TIME PERIOD (sec.)	dV θ_{dv} t	ent ent B	auto display of: ALTITUDE VELOCITY DIRECTION and FUEL
	(OPTIONAL: You may repeatedly review the output by rolling up the stack)			
	OR			
3	ENTER PERIOD FOR FREE FALL (sec)	t	A	(same as above)
4	REPEAT STEPS 3 AS NECESSARY.			
5	TO PLAY A NEW GAME , SET FLAG 1 and GO TO STEP 2.			
	NOTES: When you impact, the altitude displayed will be 0.000, and the velocity displayed will be the impact velocity.			
	The machine will display ERROR if you attempt to apply a thrust greater than 7 Gees. (69 m/sec/sec of time period).			

STEP	KEY ENTRY	KEY CODE	COMMENTS
01	*LBL B	31 25 12	THRUST
	STO C	33 13	store t
	6	06	69t
	9	09	
	x	71	dV
	r↑	35 54	fuel remaining
	RCL 0	34 00	if not enough fuel
	x ≤ y?	32 71	replace dV with
	x ↔ y	35 52	remaining fuel.
0	r ↓	35 53	is dV > 7 Gees?
	x > y?	32 81	then show ERROR.
	x	31 21	dV
	x ↔ y	35 52	subtract dV from
	r ↓	35 53	fuel supply
	STO - 0	33 51 00	X:dV_θ , Y:dV_r
	P → R	31 72	
	STO + 5	33 61 05	add to V_θ
	RCL 5	34 05	
	9	09	if V_θ < 9 m/sec.,
0	x > y?	32 81	then go to vector
	GTO 2	22 02	sum equations.
	r↑	35 54	dV_r
	STO + 4	33 61 04	add to V_r
	*LBL 1	31 25 01	COMPUTE NEW ORBIT
	CF 1	35 61 01	find $V^2=V_\theta^2+V_r^2$
	RCL 4	34 04	
	x^2	32 54	
	RCL 5	34 05	
	x^2	32 54	
	+	61	
	2	02	$V^2/2$
	/	81	
	RCL 3	34 03	Gm
	RCL 1	34 01	Gm/R
	/	81	E= $V^2/2$-Gm/R
	-	51	
	STO 6	33 06	R_6 =E
	RCL 1	34 01	
	RCL 5	34 05	R*V_θ = l
0	x	71	$R_7 \triangleq l$
	STO 7	33 07	l^2
	x^2	32 54	
	RCL 3	34 03	Gm
	/	81	$R_o = l^2/Gm$
	STO 8	33 08	store in R_8
	x	71	$E*R_o$
	2	02	$2ER_o$
	x	71	Gm
	RCL 3	34 03	$2ER_o/Gm$
0	/	81	
	1	01	SQRT(1 +$2ER_o/Gm$)
	+	61	= e
	SQRT	31 54	store in R_9
	STO 9	33 09	R_{o_R}
	RCL 8	34 08	
	RCL 1	34 01	

STEP	KEY ENTRY	KEY CODE	COMMENTS
	/	81	R_o/R
	1	01	
	-	51	e
060	RCL9	34 09	$(R_o/R - 1)/e$
	/	81	round off to make
	DSP 7	23 07	sure sum is not
	RND	31 24	greater than 1
	DSP 3	23 03	acos = dθ
	acos	32 63	
	RCL 4	34 04	determine whether
	RCL 5	34 05	spacecraft is on
	x	71	ascending or
	x > 0?	31 81	descending quad-
070	SF 2	35 51 02	rant of orbit,
	r ↓	35 53	and change sign
	F?2	35 71 02	of dθ if neces-
	CHS	42	sary.
	RCL 2	34 02	θ' =θ_i +dθ
	+	61	
	RC I	35 34	normalize to
	/	81	keep θ between
	FRAC	32 83	0 and +360.
	1	01	
080	+	61	
	FRAC	32 83	
	RC I	35 34	
	x	71	
	STO A	33 11	Store θ' in R_A
	RCL C	34 13	t
	* LBL A	31 25 11	FREEFALL PERIOD
	STO C	33 13	store t in R_C
	0	00	
	F ? 1	35 71 01	if new game, ini-
090	GTO 1	22 01	tialize orbit
	0	00	parameters
	RCL 5	34 05	if V_θ < 9 m/sec.
	9	09	go to vector-sum
	x > y?	32 81	equations.
	GTO 2	22 02	
	r ↓	35 53	V_θ
	RCL C	34 13	V_θ*t = R'$_\theta$
	x	71	Gm
	RCL 3	34 03	R^2
100	RCL 1	34 01	Gm/R^2=gravity= g
	x^2	32 54	t
	/	81	gt
	RCL C	34 13	
	x	71	gt/2
	2	02	V_r
	/	81	V_r-gt/2
	RCL 4	34 04	
	x ↔ y	35 52	(Vr-gt/2)t
	-	51	
110	RCL C	34 13	
	x	71	
	RCL 1	34 01	

REGISTERS

	1	2	3	4	5	6	7	8	9
fuel	R_i	θ_i	Gm	V_r	V_θ	E	l	R_o	e
	S1	S2	S3	S4	S5	S6	S7	S8	S9

A	B	C	D	E	I
θ'	gunfac	time	V	θ	360

STEP	KEY ENTRY	KEY CODE	COMMENTS
	+	61	$R' = R_i + (V_r - gt/2)t$
	R → P	32 72	X: R_r Y: $d\theta_i$
	r ↓	35 53	
)	STO + 2	33 61 02	$\theta_{inew} = \theta_i + d\theta$
	RCL 8	34 08	R_i
	RCL 9	34 09	e
	RCL 2	34 02	θ_i
20	RCL A	34 11	θ'
	-	51	$\theta_i - \theta'$
	cos	31 63	
	x	71	$e*\cos(\theta_i - \theta')$
	1	01	
	+	61	
	/	81	$R_i = R_o/(1+e*\cos)$
	STO 1	33 01	store in R_1
	RCL 3	34 03	Gm
	x↔y	35 52	
30	/	81	Gm/R_i
	RCL 6	34 06	E
	+	61	
	2	02	
	x	71	
	SQRT	31 54	$V_i = SQRT(2(E+Gm/R))$
	STO D	33 14	store in R_d
	RCL 1	34 01	R
	x	71	R*V
	RCL 7	34 07	1
40	x↔y	35 52	
	/	81	1/RV
)	DSP 7	23 07	round off to less
	RND	31 24	than 1 or -1
	DSP 3	23 03	restore display
	acos	32 63	$\theta_v = \arccos(1/RV)$
	RCL 7	34 07	1_v
	RCL 2	34 02	θ_i
	RCL A	34 11	θ'
	-	51	$d\theta$
50	sin	31 62	if 1&sin($d\theta$) are
	x	71	of different sign,
	x<0 ?	31 71	change sign of
	SF 2	35 51 02	θ_v to correct
	r ↓	35 53	quadrant of orbit
	F ? 2	35 71 02	
	CHS	42	
	* LBL 3	31 25 03	OUTPUT
	STO E	33 15	store θ_v
	RCL D	34 14	V
60	P → R	31 72	X: V_θ Y: V_r
	STO 5	33 05	
	r ↓	35 53	store velocity
	STO 4	33 04	components
	RCL 1	34 01	R_i
	RCL B	34 12	R_i
	-	51	surface
	x<0 ?	31 71	find impact for
	GSB 4	31 22 04	negative alt.
)			

STEP	KEY ENTRY	KEY CODE	COMMENTS
	RCL D	34 14	V
170	RCL E	34 15	θ_v
	RCL 0	34 00	fuel supply
	STACK	32 84	display status
	R/S	84	
	*LBL 2	31 25 02	VECTOR SUMS
	r ↓	35 53	
	r ↓	35 53	
	2	02	
	/	81	$dV_\theta/2$
	CHS	42	
180	RCL 5	34 05	
	+	61	$V_{\theta new} - dV_\theta/2 = V_\theta$
	RCL C	34 13	t
	x	71	$V_\theta * t = R'_\theta$
	x↔y	35 52	dV_r
	RCL 3	34 03	Gm
	RCL 1	34 01	R
	x^2	32 54	
	/	81	$g = gravity = Gm/R^2$
	RCL C	34 13	t
190	x	71	g*t
	-	51	$dV_r - g*t = V_{r\,new}$
	STO + 4	33 61 04	
	2	02	
	/	81	(delta V_r)/2
	CHS	42	
	RCL 4	34 04	$V_{rnew} - (\text{delta } V_r)/2$
	+	61	t
	RCL C	34 13	delta R
	x	71	
200	RCL 1	34 01	
	+	61	$R_{r\,new}$
	R → P	32 72	X: R_i Y: $d\theta_i$
	STO 1	33 01	
	x↔y	35 52	
	STO + 2	33 61 02	
	*LBL 0	31 25 00	
	RCL 4	34 04	V_r
	RCL 5	34 05	V_θ
	R → P	32 72	X: V Y: θ_v
210	STO D	33 14	
	x↔y	53 52	
	GTO 3	22 03	output results
	*LBL 4	31 25 04	IMPACT
	STO - 1	33 51 01	altitude = 0
	3	03	$3 \approx 2*g$
	x	71	2g*altitude
	RCL 4	34 04	V_r
	x^2	32 54	
	+	61	$V_r^2 + 2ga$
220	ABS	35 64	
	SQRT	31 54	$V_{rzero} = \sqrt{(V^2+2ga)}$
	CHS	42	new V_r
	STO 4	33 04	
	GTO o	22 00	

LABELS

A	B	C	D	E
freefall	thrust			
a	b	c	d	e
0 unused	1 initial	2 vector	3 output	4 impact

FLAGS

0	
1	initial
2	used

SET STATUS

FLAGS	ON	OFF
0	☐	☒
1	☒	☐

TRIG	
DEG	☒
GRAD	☐

DISP	
FIX	☒
SCI	☐