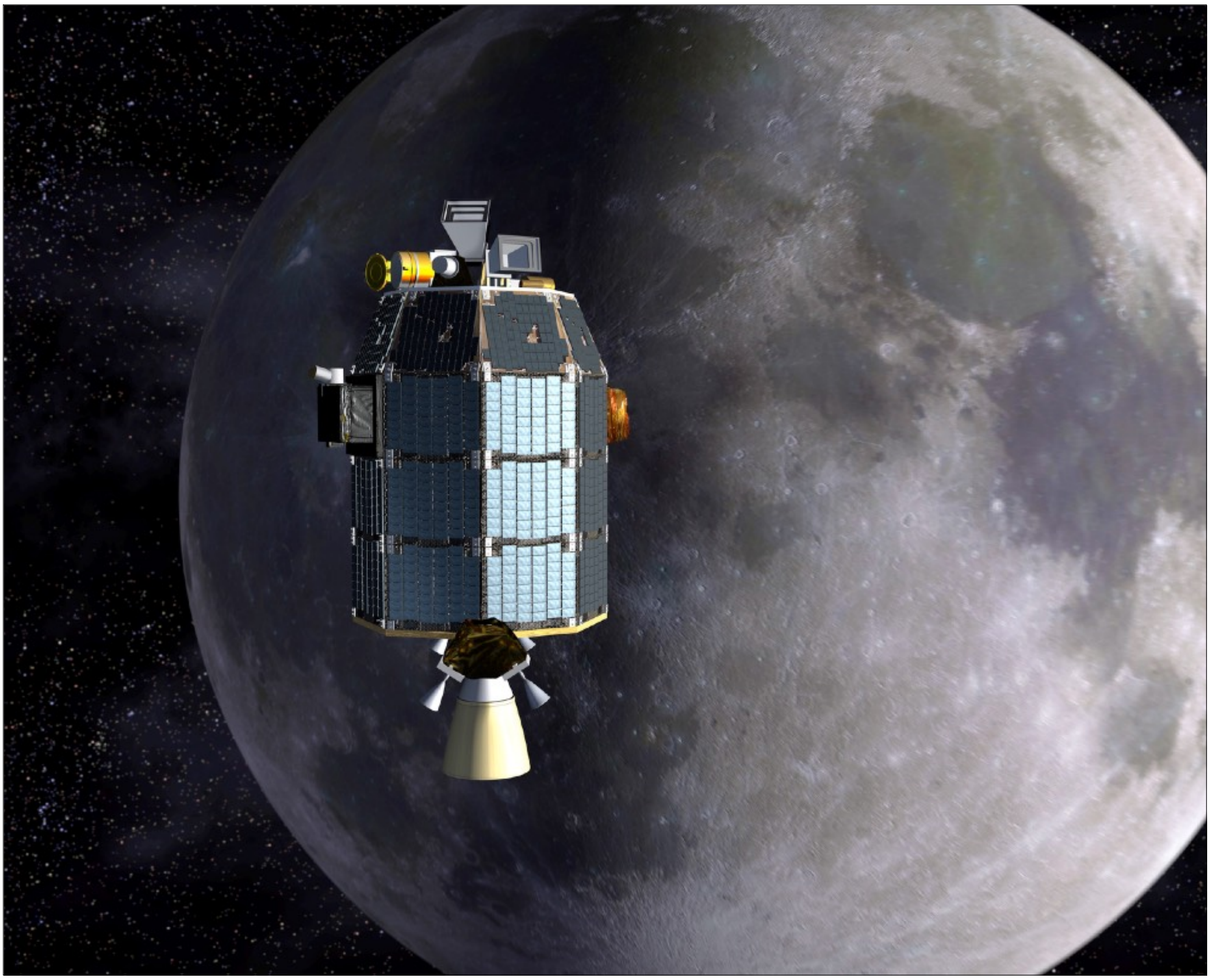


Progetto Astrodinamica - Lunar Free Return Orbit

Preparato per: E. Lorenzini, G. Colombatti
Preparato da: G. Boscaro, R. Cagnato, R. Frezza, E. Rossetto

21 maggio 2018

RIEPILOGO ESECUTIVO

Scopo

Lo scopo del progetto è realizzare un'orbita di trasferimento Terra-Luna. L'orbita deve garantire il ritorno a Terra dopo il flyby lunare. Il perigeo dell'orbita di rientro deve essere compreso in una fascia tra i 300 Km ed i 1000 Km di altezza. L'orbita di parcheggio iniziale è circolare e ad un'altezza fissata di 400 Km.

Obiettivi

L'obbiettivo del progetto è dimensionare un'orbita che rispetti tutti i requisiti ed inoltre ottimizzare il ΔV correttivo che potrebbe essere necessario all'arrivo alla sfera d'influenza della Luna per permettere l'effettivo ritorno a Terra dopo il flyby.

Soluzione

La soluzione da noi adottata prevede di non utilizzare alcun ΔV correttivo. L'unico impulso sarà quindi dato alla partenza dalla Terra. L'obbiettivo sarà quindi minimizzare il ΔV di *burn-out*. Inoltre, in previsione di un possibile sbarco nella superficie lunare, si cercherà un'orbita con un periselenio abbastanza basso, nell'ordine dei 100Km di altezza.
La soluzione inoltre assume le seguenti ipotesi semplificative:

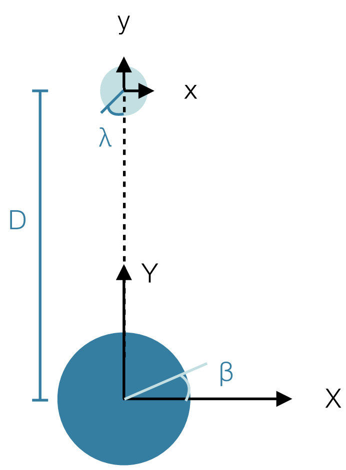
- Orbita della Luna circolare (e = 0) e complanare all'equatore terrestre
- Nessuna perturbazione orbitale
- ΔV impulsivi
- Satellite puntiforme

L'orbita sarà progettata in MatLab. I dati raccolti verranno poi analizzati in Excel e l'orbita scelta verrà visualizzata su un grafico MatLab.

Struttura del progetto

- **Introduzione**

Prima di passare all'effettiva esposizione del progetto è necessario introdurre alcune informazioni fondamentali. Innanzitutto, per lo svolgimento sono stati utilizzati due sistemi di riferimento. Il primo è geocentrico, con gli assi X,Y ortogonali e nel piano equatoriale. Il secondo sistema è centrato invece sulla Luna e verrà utilizzato per computare l'orbita iperbolica di flyby. Questo sistema è definito come una traslazione di D lungo l'asse Y geocentrica. I due assi y sono definiti in moto che essi siano allineati nell'istante in cui il satellite arriva alla sfera di influenza lunare. Il centro di questo sistema traslerà nel tempo con un moto circolare uniforme seguendo l'orbita



lunare circolare. Esso inoltre non subirà alcuna rotazione. Gli assi saranno sempre paralleli agli assi geocentrici. Due parametri fondamentali, visibili nella figura precedente, sono gli angoli β e λ. L'angolo β rappresenta l'angolo (rispetto al sistema geocentrico) a cui si effettua il burn-out. L'angolo λ è l'angolo, rispetto all'asse y negativo lunare, del raggio selenocentrico all'arrivo del satellite nella sfera d'influenza lunare.

- **Procedimento**

Si è scelto di dividere l'intera orbita in tre fasi distinte seguendo il metodo delle *patched conics*. Le fasi saranno: partenza, flyby, ritorno.
L'orbita di partenza viene calcolata risolvendo il problema di Lambert. Dati t (*Time of Flight*) e posizione iniziale e finale si ottengono le velocità di partenza e arrivo (alla sfera di influenza della Luna) in coordinate geocentriche. La posizione iniziale (o di *burn-out*) sarà data da $[R_0 cos(\beta), R_0 sin(\beta), 0]$ mentre quella finale sarà $[R_1 cos(\gamma), R_1 sin(\gamma), 0]$ dove:

$$R_1 = \sqrt{D^2 + R_S^2 - 2DR_S cos(\lambda)}$$

$$\gamma = arcos\left(\frac{D - R_S cos(\lambda)}{R_1}\right)$$

$$R_s = R_e \left(\frac{1}{81}\right)^{\frac{2}{5}} = 66212Km$$

$$R_0 = 400Km$$

$$R_e = 6378Km$$

t, β, λ saranno i tre parametri iterativi del processo. Al variare di essi otterremo varie orbite. Alcune non raggiungeranno la Luna, alcune si schianteranno sulla superficie lunare, alcune potrebbero indirizzarsi verso il sistema solare esterno, altre torneranno verso la Terra ma non entro i limiti fissati. Alcune orbite, però, rispetteranno tutti i requisiti richiesti e verranno quindi salvate in un file. Tra queste verrà scelta l'orbita finale.
La soluzione al problema di Lambert ci restituisce le velocità necessarie per arrivare alla Luna in quel determinato tempo. Dalla velocità V_0 si ottiene facilmente il ΔV_{bo} (V_p velocità circolare di parcheggio):

$$\bar{V}_p = \sqrt{\frac{\mu_E}{R_0}} \begin{pmatrix} sin(\beta) \\ cos(\beta) \\ 0 \end{pmatrix} \rightarrow \Delta \bar{V}_{bo} = \bar{V}_0 - \bar{V}_p$$

Conosciamo inoltre la velocità V_1 di arrivo alla sfera d'influenza e la posizione geocentrica del satellite all'arrivo. È necessario però avere questi dati in coordinate lunari per procedere all'elaborazione dell'orbita di flyby. Ricordando come era stato definito il sistema di riferimento lunare risulta ovvio che:

$$V_m = \sqrt{\frac{\mu_E}{R_m}} \rightarrow \bar{v}_\infty = \bar{V}_1 - [-V_m,\ 0,\ 0]$$

$$\bar{R}_1^{(M)} = \bar{R}_1^{(GEO)} - [0,\ D,\ 0]$$

A questo punto si hanno tutte le condizioni iniziali per elaborare l'orbita di flyby. Si è scelto di propagare nel tempo l'orbita utilizzando l'equazioni di Keplero scritta in termini di anomalia universale e coefficienti di Lagrange. Per farlo basta discretizzare il tempo e calcolare posizione e velocità del satellite ad ogni istante. Se il satellite si schianta sulla superficie, l'orbita viene scartata. Viceversa, il satellite prosegue col flyby fino al raggiungimento della sfera d'influenza dall'altro lato della Luna. Le condizioni finali di quest'orbita iperbolica saranno le condizioni iniziali dell'orbita di ritorno. Nel frattempo però la Luna ha proseguito nel suo percorso attorno alla Terra, per cui il sistema di riferimento lunare non sarà più allineato con l'asse y dell sistema geocentrico. Per proseguire sarà quindi prima necessario calcolare le nuove coordinate geocentriche del satellite alla fine del flyby lunare.

$$w_m = \frac{V_m}{R_m} \rightarrow \theta = w_m \cdot t_{flyby}$$

$$\bar{R}_2^{(GEO)} = D \begin{pmatrix} -sin(\theta) \\ cos(\theta) \\ 0 \end{pmatrix} + \bar{R}_2^{(M)}$$

$$\bar{V}_2^{(GEO)} = \begin{pmatrix} -V_m sin(\frac{\pi}{2}) - \theta \\ -V_m cos(\frac{\pi}{2}) - \theta \\ 0 \end{pmatrix} + \bar{v}_\infty$$

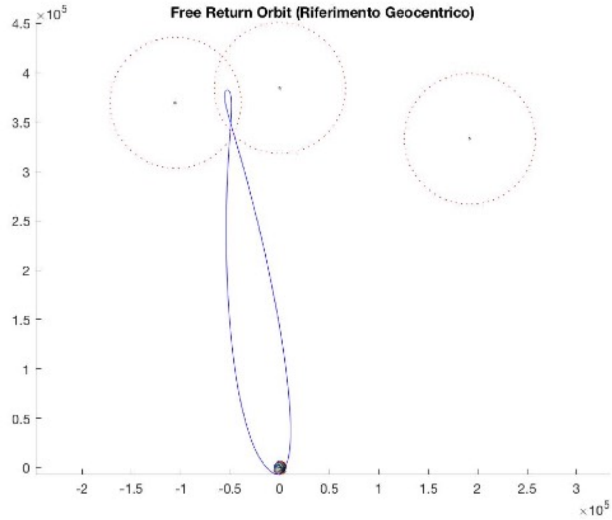
Anche l'orbita di ritorno è una propagazione nel tempo con Keplero come visto precedentemente. Se il satellite si allontana dalla Terra, oppure avvicinandosi ad essa impatta con la superficie, l'orbita viene scartata. Anche se l'orbita è valida, però, bisogna fare attenzione che il perigeo sia entro la fascia richiesta. Solo in questo caso l'orbita viene salvata.

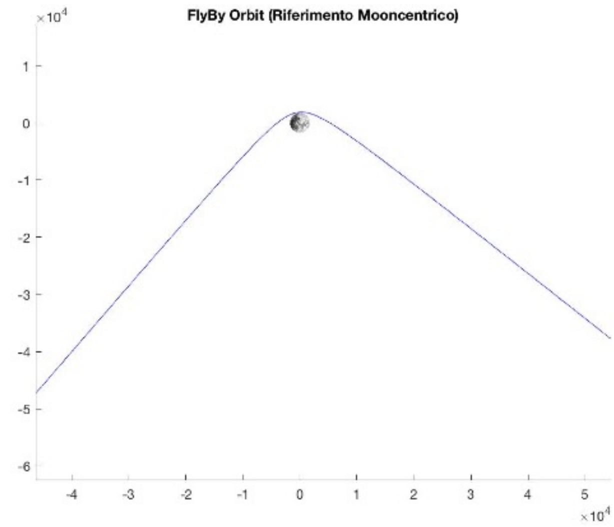
Il ciclo che elabora le orbite itera su tutte le combinazioni possibili dei tre parametri entro degli intervalli fissati. Per quanto riguarda il *ToF,* esso non si discosterà molto dalle tipiche 72h delle missioni *manned* dirette alla Luna. Per β e λ non ha sicuramente senso andare oltre i ± 90°. Con una prima esecuzione del codice,

impostando intervalli di valori molto ampi si possono ricavare velocemente dei risultati, molto grezzi, che aiutano però già a scartare molti di questi angoli, i quali richiederebbero ΔV troppo grandi oppure non permetterebbero mai di raggiungere la Luna. Si procede poi all'analisi dei risultati, verificando gli intervalli di valori per cui si hanno le orbite migliori. Si ripetono le simulazioni sui nuovi intervalli trovati, sempre più stretti, in cui i valori sono suddivisi con passi sempre più piccoli.
Tutte le orbite di successo vengono salvate in un file Excel accompagnate dai parametri più importanti per la selezione dell'orbita finale, tra cui il ΔV e l'altezza del periselenio. Si analizzano quindi i risultati finali e si sceglie l'orbita più adatta alle esigenze.
Nel nostro caso l'orbita scelta ha i seguenti parametri:

- ΔV_{bo} = 3.14 Km/s
- *ToF* = 137h (70h al periselenio)
- $H_{periselenio}$ = 102 Km
- β = 23°
- λ = 44.4°

che è stata scelta avendo uno tra i ΔV più bassi ed l'altezza di flyby più vicina a quella ottimale per l'eventuale l'atterraggio. Tra i risultati vi erano alcune orbite notevoli che potrebbero far risparmiare fino a 40 m/s di ΔV ma che avrebbero un'altezza di periselenio troppo alta (nell'intorno di 550 Km). Queste orbite sono da considerarsi migliori di quella scelta nel caso l'obiettivo della missione fosse un semplice flyby senza atterraggio.
Infine, dopo aver scelto l'orbita, questa può essere visualizzata in un grafico grazie a MatLab (*vedi figure a destra*). Oltre ad un grafico statico, è possibile anche visualizzare un'animazione dell'orbita dello *spacecraft* dal lancio fino al ritorno sulla Terra (*vedere video allegato al documento*).





- **Software**

Di seguito verranno discussi i passi più significativi del codice MatLab scritto per il progetto ed i dettagli più tecnici.

- Propagazione Keplero

```
alphaM = 2/norm(R1_M) -norm(vinf_in)^2/muM;
t_flyby = 60; dt = 1200;

while(norm(R) < Rs && i < itmax_M)

    i = i+1;

    x = kepler_U(t_flyby, norm(R1_M), vr1, alphaM);
    [f,g] = f_and_g(x, t_flyby, norm(R1_M), alphaM);
    R = f*R1_M + g*vinf_in;

    if(norm(R) < Rm)
        disp('Impatto con la superficie LUNA');
        impatto_luna = true;
        break;
    end

    vetR(i) = norm(R);
    vetRx_M(i) = R(1);
    vetRy_M(i) = R(2);
    vetDD(i) = Wm*t_flyby;

    if(i > 1 && norm(vetR(i))+abs(norm(vetR(i))-norm(vetR(i-1)))+fR_M > Rs)
        dt = 60;
    end
    t_flyby = t_flyby + dt;
end
```

Questo ciclo propaga l'orbita iperbolica selenocentrica. Per propagare con Keplero è necessario discretizzare il tempo, per cui si imposta un *dt* inizialmente di 20 minuti. Il ciclo continua ad iterare finché non raggiungo il raggio della sfera d'influenza lunare oppure supero le iterazioni massime consentite. La funzione *kepler_U* restituisce l'anomalia universale date le condizioni iniziali (raggio e velocità radiale), il reciproco del semiasse maggiore dell'orbita (*alpha*, ricavato dall'equazione dell'energia) e il tempo trascorso dall'istante iniziale. Data l'anomalia universale si ricavano poi i coefficienti di Lagrange e di conseguenza il vettore posizione all'istante *t.* Se la norma di questo raggio risulta più piccola del raggio della Luna il ciclo viene interrotto e viene segnalato l'impatto. Se cosi non fosse, il vettore posizione e le sue componenti vengono salvate per uso futuro. Viene inoltre salvato l'angolo che la Luna ha percorso rispetto al sistema geocentrico nel tempo *t*. Essendo il tempo discretizzato con un passo di 20 secondi, il satellite percorre più di 1000 Km ogni ciclo. Potrebbe essere quindi che il ciclo si interrompa quando lo *spacecraft* sia già molti km fuori dalla sfera d'influenza. Un *if* all'interno del ciclo controlla se al ciclo successivo c'è il rischio di uscire dalla sfera d'influenza (*fR_M* è un fattore di sicurezza). Se cosi fosse il passo *dt* viene ridotto a 60s in modo da essere più precisi. Se il passo fosse stato cosi piccolo fin dall'inizio i tempi di esecuzione del programma si sarebbero allungati di molto.

Il ciclo che propaga l'orbita di ritorno è impostato allo stesso modo ed è praticamente uguale se non per questo dettaglio in più:

```
if(norm(R) > prec)
       break;
end
```

che controlla se lo *spacecraft* ha raggiunto il suo perigeo. Se la norma del raggio geocentrico del satellite è maggiore del suo valore al ciclo precedente vuol dire che lo *spacecraft* ha raggiunto il suo minimo è sta cominciando ad allontanarsi dalla Terra. Per cui se questa condizione è verificata il ciclo viene fermato. Si verificherà poi che questo perigeo sia entro i limiti imposti. Se ci si volesse immettere in un'orbita di parcheggio attorno alla Terra per poi rientrare sarebbe necessario un ΔV in questo punto.

- Salvataggio dati

```
if(hp_finale >= 300 && hp_finale <= 1000 && not(impatto_terra) && not(impatto_luna))

       n_successi = n_successi + 1;
       M_Risultati{n_successi,1} = t_andata;
       M_Risultati{n_successi,2} = beta;
       M_Risultati{n_successi,3} = lambda1;
       M_Risultati{n_successi,4} = V0;
       M_Risultati{n_successi,5} = alphaE;
       M_Risultati{n_successi,6} = vetDD;
       M_Risultati{n_successi,7} = vetRx_M;
       M_Risultati{n_successi,8} = vetRy_M;
       M_Risultati{n_successi,9} = vetRx_ret;
       M_Risultati{n_successi,10} = vetRy_ret;
       M_Risultati{n_successi,11} = hp_finale;
       M_Risultati{n_successi,12} = hp_M;
       M_Risultati{n_successi,13} = deltaV;
       M_Risultati{n_successi,14} = t_andata+t_flyby+t_ritorno;
       M_Risultati{n_successi,15} = t_andata + t_hp;
end
save('data_results.mat', 'M_Risultati');
```

Viene verificato se l'orbita è valida tramite un *if*, che verifica che il satellite non abbia impattato con la superficie dei corpi celesti e che l'altezza finale sia entro i limiti imposti dai requisiti. I dati vengono salvati in un *cell array*, che a differenza dei vettori e matrici standard di MatLab, permette di salvare qualsiasi oggetto[1] nelle sue celle. Questo ci permette di salvare facilmente tutti i vettori delle posizioni assunte dal satellite nel tempo oltre ad altri dati necessari per graficare successivamente l'orbita (*alpha,* V_0). Inoltre vengono salvati tutti i dati utili per selezionare l'orbita finale, come ΔV, l'altezza del periselenio ed tutti i tempi di percorrenza.

[1] raccolta/struttura di dati di qualsiasi tipo

- Esportazione Excel

I dati vengono esportati su Excel col seguente codice

```
header=["Tempo Andata","Beta","Lambda1","V0","H Perigeo","H Periselenio","Delta V",'Tempo Totale','Tempo Periselenio'];
X = [header;X];
xlswrite(strcat(nomefile,'.xls'),X);
```

dove X è la matrice di dati che era stata salvata precedentemente. A questa viene aggiunta una riga in testa con i titoli delle colonne, in modo da rendere più leggibile il foglio Excel.
Il comando che effettivamente esporta la matrice nel foglio di calcolo è *xlswrite* che richiede in ingresso il nome del file con l'estensione *.xls* e la matrice dei dati.

- Grafici e animazioni

Per visualizzare i grafici, vengono semplicemente disegnati in una figura tutti i vettori posizione salvati precedentemente, rappresentanti tutte le posizioni occupate dal satellite nella sua orbita. Vengono inoltre disegnati i corpi celesti utilizzando delle immagini della Terra e della Luna.

```
img_earth = imread('img_earth.jpg'); imagesc([-Re Re], [-Re Re],img_earth);
```

Le funzioni utilizzate sono *imread,* che permette di caricare l'immagine e salvarla in una variabile. Il comando *imagesc* disegna l'immagine nel grafico. I vettori indicano le coordinate del vertice in basso a sinistra e l'altezza e larghezza dell'immagine a partire da esso.
Per produrre le animazioni si utilizza invece *kepler_U* discretizzando il tempo. I cicli sono impostati sulla falsariga dei cicli utilizzati per propagare le orbite durante l'ottimizzazione. In questo caso è necessario propagare anche l'orbita di andata. Infatti se si risolvesse il problema di Lambert si avrebbero solo posizione iniziale e finale, non l'intera orbita.
Per mostrare l'evoluzione nel tempo dell'orbita bisogna aggiungere all'interno di ogni ciclo il comando *pause(t).* Questo comando permette di introdurre un *timeout di t* secondi ad ogni iterazione, permettendo di creare l'animazione. Se non si inserisse questo *timeout,* MatLab elaborerebbe il ciclo a velocità macchina per cui l'animazione non verrebbe visualizzata e apparirebbe solo la posizione finale.