参赛队员姓名：沈含岳

中学：上海世界外国语中学国际部

省份：上海市

国家/地区：中国，华东

指导教师姓名：陈浦胤，李云飞

指导教师单位：上海应用技术大学，上海宇航系统工程研究所

论文题目：一种基于太阳帆的地球-木星行星际转移轨道优化设计

一种基于太阳帆的地球-木星行星际转移轨道优化设计

沈含岳

2022 年 9 月 13 日

摘要

太阳帆利用光压产生的推力完成不消耗任何额外燃料的连续小推力飞行，是一种极具发展潜力且经济的行星际飞行方式，尤为适合长期行星际任务，也可用于其它多种任务。本文在基于一定简化条件的情况下，通过建立太阳帆航天器的日心动力学模型，采取数值求解，分段建模，组合优化的设计方式，进行分为三段的地球-火星-木星序列的借力行星际转移轨道设计。通过数值计算，得出了太阳帆在满足时间限制的情况下，相较脉冲、电推进等传统推进方式可携带更多的有效载荷，充分体现出了太阳帆不消耗任何化学燃料的优势，为长期的行星际转移任务提供了一种新思路，即利用转移时间的增加换取极大的有效载荷。不同于前人对于一整段转移轨道进行全局优化的方法，本文采用了借力和分段与局部最优化的组合，一定程度上减少了计算量，且可以扩充更多分段，便于进一步改进设计。

关键词：星际飞行，太阳帆，轨道设计

目录

1 变量与常数

轨道和力学:

a: 半长轴,

e: 偏心率,

r: 轨道半径，单位km,

p: 半正焦弦,

E: 偏近点角,

f: 真近点角,

h: 角动量,

μ: 重力系数,

δ: 借力偏转角,

γ: 飞行路径角,

ΔV: 速度增量.

航天器（太阳帆）系数:

β: 轻度系数,

α: 俯仰角(太阳帆方向),

δ: 时钟角,

C_{rf}: 反射率,

σ: 负载系数,

I_{sp}: 比冲,

ρ: (面)密度.

常数:

太阳引力常数$\mu = 132.712\text{e}9\text{km}^3/\text{s}^2$,

火星引力常数$\mu_M = 42828\text{km}^3/\text{s}^2$,

地球轨道速度$v_E = 29.78\text{km/s}$,

火星轨道速度$v_M = 29.78\text{km/s}$,

木星轨道速度$v_J = 13.06\text{km/s}$,

地球轨道半径$r_E = 149.598\text{e}6\text{km}$,

火星轨道半径$r_M = 227.956\text{e}6\text{km}$,

木星轨道半径$r_J = 778.279\text{e}6\text{kmfi}$

天文单位$1\text{AU} = 149.598\text{e}6\text{km}$.

负载系数为太阳帆面积与航天器总质量的比值，体现出了帆的相对大小。面密度为太阳帆质量（薄膜加上结构质量）与其面积之比，体现出了太阳帆材料的性能。

2 引言

木星作为距离地球最近的气态巨星，是人类不断探索的目标。1972年3月2日发射的先驱者10号探测器成为了首个近距离观测木星的探测器，与当年12月传回相关图像。继旅行者号飞跃木星后，美国先后发射了伽利略号和朱诺号木星轨道器，其中伽利略号在其核燃料耗尽后已坠毁于木星大气层中。这些探测器都使用化学或核动力推进器，因此在行星际转移途中消耗大量燃料。尽管有行星借力序列（如EVEEJ, EVEMJ)的帮助，但是其有效载荷比(剩余质量与起飞质量之比）较小。对于这些长期任务，增加航天器的有效载荷是至关重要的问题。

为了减少燃料，采取一种新颖的推进方式是一种选择。而利用电磁场的动量，即光压，的太阳帆是一种符合逻辑的选择，因为它除了转向使用的电能，不会消耗任何额外的能量，进而也不会消耗推进剂。

太阳帆的想法由来已久。从齐奥尔科夫斯基的最初想法开始[1]，到日本的IKAROS飞行器和Planetary Society发射的LightSail 2卫星，先辈们已经从理论和实验中验证了太阳帆作为一种不使用燃料的推进方式的可行性，在行星际探测领域和日心轨道中均有重要的应用，具体有月球、火星等探测任务[2][3], 亦有H-Reveral Trajectory这类周期性轨道可用于不同种类的任务[4],甚至可用于远日行星或小行星的飞跃任务[4].太阳帆虽然有转移时间较长的缺点，但可以一定程度上减少传统推进方式带来的燃料消耗，从而达到携带更多有效载荷的目的。

从对文献的阅览中可发现，利用太阳帆的任务，除周期性日、地轨道外，行星际探测轨道多为仅交会的轨道，尤其对木星等远日行星仅仅掠过而非使航天器入轨。随着人们对于太空的探索不断进步，探索太阳系起源等问题，探索远日行星必将成为未来的发展趋势之一，而为了这个目的，使轨道器环绕行星进行长时间的观测遥感是有必要的。因此，本文将着手于利用太阳帆的局部优化控制进行组合，同时运用行星借力，实现航天器地木转移，且使航天器到达木星时剩余速度（即与木星相对速度）尽可能小（也就意味着入轨需要的燃料少，有效载荷大）。

为便于体现加入太阳帆对于任务的影响，本文任务采用与《行星际飞行理论与应用》[6]一书中木星任务设计相仿的任务参数。书中优化设计有使用脉冲推进器的情况下航天器进入$5R_J \times 10R_J$的木星轨道，其中R_J为71492千米(木星平均半径), 木星探测器的初始总质量为5000kg，且配有比冲为450s的化学推进系统(本任务中用于制动)，文中航天器初始质量与化学推进系统比冲同该书。该书中也描述了连续小推力转移的方案，地球逃逸速度增量为0km/s，任务计算至航天器到达木星引力影响球，给出的三种方案到达木星的剩余速度均为0km/s。由于太阳帆属于一种连续小推力推进方式，优化时结果将参照其它连续小推力方案得到的结果比对。对于初始轨道方面，本文中将采用火箭直接将载荷送入行星际转移轨道，而非先进入停泊轨道，火箭的发射能量C3取$0\text{km}^2/\text{s}^3$(此数值小于书中给出的脉冲推进探测轨道设计中的数值,其中最优化后分别有14.565, 10.085

18.649[6]三个数值，与书中所有连续小推力的方案初始条件相同)，即航天器离开地球的速度增量为0km/s。航天器任务终止条件为到达木星引力影响球且剩余速度为0km/s。

假设太阳帆薄膜反射率为 0.90[7]，面密度为 $1g/m^2$[8](多方面材料显示，尽管符合条件的材料仍处于试验阶段，但此面密度是长期任务所需要的)，太阳帆轻度系数待定，为优化变量之一。

3 模型假设

为方便研究模型，做出以下假设:

(1) 假设所有涉及的行星都在围绕太阳的共面圆轨道上运行，圆轨道的轨道半径取其实际轨道的半长轴, 星体都为质地均匀的完美球体。

(2) 不考虑行星大气层对于航天器的影响，同时也不考虑星体摄动对轨道造成的影响,即行星的引力在且仅在行星希尔半径内才会影响航天器。

(3) 在太阳帆进行行星际转移的过程中，不考虑行星对于太阳光的遮掩，同时航天器所受到的所有光压均来自太阳。

(4) 当航天器靠近行星希尔半径时，假设它可以从任意的角度进入行星引力范围。

(5) 假设太阳帆的推进效率为百分之百，且其性能在任务中没有变化，不随着时间、温度等条件而老化，且太阳帆转向、收起、展开的时间可以忽略不计，太阳帆的推力没有损耗，航天器飞越借力掠过的行星所用的时间也可以忽略不计。

(6) 假设航天器所有行星际转移轨道均为顺行轨道。

*由于仅考虑平面情形，航天器的时钟角δ在整个转移过程中都设为$\frac{\pi}{2}$。

4 分析与建立模型

4.1 航天器的力学模型

太阳帆，顾名思义由太阳驱动，其物理原理为电磁场的动量，即“光压”，所产生的微弱推力（不是其它高能粒子）。其受力如简图所示：

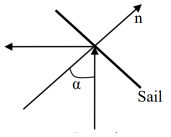

图 1: 航天器光压受力原理图[9]。

定义“有效载荷”为航天器消耗化学燃料以后剩余的质量减去太阳帆薄膜的质量,反映了它可搭载的科学仪器的多少。此次任务中，推进器只在木星轨道插入时点火，通

过齐奥尔科夫斯基方程即可求出某一速度增量对应需要的燃料消耗。

$$m = m_i \cdot \exp(-\frac{\Delta V}{I_{sp}g}), \tag{1}$$

其中$g = 0.0098\ \mathrm{km/s^2}$。$m$为推进器点火后质量，$m_i$为点火前质量。

太阳帆的质量由太阳帆表面积和太阳帆面密度确定。建立完整的模型也需要太阳帆的一系列重要参数。记航天器在1AU处受到的光压为P_0，反射率C_{rf}为0.90:

$$P_0 \approx 4.56(1 + C_{rf}) = 8.664\mu\mathrm{Pa}. \tag{2}$$

由此可得任意位置航天器受到的光压，它与航天器到太阳的距离成平方反比。r为航天器与太阳的距离，此公式中单位为AU：

$$P \approx 4.56(1 + C_{rf}) \cdot \frac{1}{r^2}\mu\mathrm{Pa}. \tag{3}$$

其中4.56μPa为1AU处的太阳光压。

轻度系数β是确定太阳帆工作效率的重要性能指标，定义为特征负载系数与负载系数的比值，从它与特征负载系数，可以进而得出太阳帆的一系列物理参数，包括其面积与质量。特征负载系数的具体推导由《Solar Sailing: Technology, Dynamics, and Mission Applications》[10]给出，书中反射率假设为1，此处改为任意反射率下的公式：

$$\begin{aligned}
\sigma^* &= 1.53 \cdot (1 + C_{rf})/2000, \\
\sigma &= \frac{\sigma^*}{\beta} \\
A_{sail} &= 5000kg/\sigma, \\
M_{sail} &= \rho_{sail} \cdot A_{sail}.
\end{aligned}$$

特征加速度也是衡量太阳帆性能的重要指标，指在1AU处，太阳帆获得的最大的加速度(即$\alpha = 0$)，一些著作中会将它作为太阳帆的优化变量。由特征加速度可求出任意位置，任意角度太阳帆贡献的加速度:

$$\begin{aligned}
a^* &= \frac{P_0}{\sigma}, \\
a &= \frac{P\cos^2(\alpha)\boldsymbol{n}}{\sigma} = \frac{\beta\mu}{r^2}\cos^2(\alpha)\boldsymbol{n},
\end{aligned} \tag{4}$$

其中向量$\boldsymbol{n}$为太阳帆法向方向向量，也就是太阳帆推力方向上的单位向量。关于加速度的得到的详细推导，参见《Principals of Solar Sailing》[9]。

通过上述公式，可以推导出航天器的运动方程，即万有引力方程加上一个力:

$$\ddot{\boldsymbol{r}} + \frac{\mu}{r^3}\boldsymbol{r} = \frac{\beta\mu}{r^2}\cos^2(\alpha)\boldsymbol{n}. \tag{5}$$

描述太阳帆方向的角度α可以在正负90°的范围内任意取值(实际情况下可能会由于种种限制而取一个略小区间)，而太阳帆的操控方式直接决定了它的飞行路径。

若使太阳帆在转移中始终正对太阳，即$\alpha = 0$, $\boldsymbol{n}$与$\boldsymbol{r}$方向相同，则方程(5)简化为：

$$\ddot{\boldsymbol{r}} + \frac{\mu(1-\beta)}{r^3}\boldsymbol{r} = 0, \tag{6}$$

此情况下航天器轨迹仍是圆锥曲线，太阳帆的作用表现在减小了太阳的引力系数。可设有效引力系数μ_e为$\mu(1-\beta)$,方程(6)可进一步简写为：

$$\ddot{\boldsymbol{r}} + \frac{\mu_e}{r^3}\boldsymbol{r} = 0, \tag{7}$$

角α在取另一些函数时，可形成螺旋轨道等轨道，后文中将对此进行讨论。

4.2 太阳帆最大推力

优化太阳帆的轨迹时，将一指定方向上的加速度最大化尤为重要，特别是因为太阳帆所提供的推力受限于其物理原理，故只能朝向“外”推进，不能像常规的推进器提供任意方向的推力。

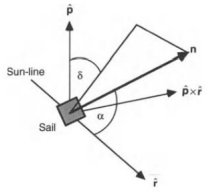

图 2: 太阳帆相关各个角度的几何关系图

图2由McInnes[10]所著一书给出，其中$\boldsymbol{p}$为垂直于轨道平面的向量。

太阳帆提供的推力在任意方向$\boldsymbol{k}$(单位向量)上的分量为:

$$F_k = 2PA_{sail}\cos^2(\alpha)(\boldsymbol{n}\cdot\boldsymbol{k}), \tag{8}$$

设α_f为$\boldsymbol{k}$与径向的夹角，则可以将上式化为:

$$F_k = 2PA_{sail}\cos^2(\alpha)(\cos(\alpha)\cos(\alpha_f) + \sin(\alpha)\sin(\alpha_f)), \tag{9}$$

若要求得某个太阳帆朝向α,使得其能在某给定方向上提供最大的推力，则将F_k对α求偏导，令$\frac{\partial F_k}{\partial \alpha} = 0$得到:

$$2\sin(\alpha)\cos(\alpha-\alpha_f) + \cos(\alpha)\sin(\alpha-\alpha_f) = 0,$$

解得：

$$\alpha = \tan^{-1}\left(\frac{\sqrt{9+8\tan^2(\alpha_f)}-3}{4\tan(\alpha_f)}\right). \tag{10}$$

确定了需要的方向后，将其代入上述公式即可获得相应的最优控制方法。

4.3 仿真动力模型

上文阐述了太阳帆的力学模型和结论的构建。在使用Matlab进行数值积分时，，不同于描述物理规律使用的极坐标方程，采用笛卡尔坐标系建立模型，因此使用如下方程组描述运动模型以及相关的轨道根数:

$$
\begin{aligned}
r &= \sqrt{x^2+y^2},\\
p &= \frac{1}{\mu}(x\dot{y}-y\dot{x})^2,\\
a &= (\frac{2}{r}-\frac{\dot{x}^2+\dot{y}^2}{\mu})^{-1},\\
e &= \sqrt{1-\frac{p}{a}},\\
f &= \cos^{-1}(\frac{1}{e}(\frac{p}{r}-1)),\\
\ddot{x} &= -\frac{\mu}{r^3}x+\frac{\beta\mu}{r^3}\cos^3(\alpha)x-\frac{\beta\mu}{r^3}\cos(\alpha)\sin(\alpha)y,\\
\ddot{y} &= -\frac{\mu}{r^3}y+\frac{\beta\mu}{r^3}\cos^3(\alpha)y+\frac{\beta\mu}{r^3}\cos(\alpha)\sin(\alpha)x,\\
\alpha &= F(a,e,f).
\end{aligned}
\tag{11}
$$

这样的模型最大限度规避了使用角度，将轨道根数也用笛卡尔坐标下的状态表达出来，便于计算、绘图等操作。对模型对应的m文件使用ode数值积分器即可作为太阳帆任务的积分器。

4.4 局部最优控制方法

局部最优控制，即对于某一特定参数进行最优化讨论。为了探讨局部最优策略，先改用高斯无奇点卫星运动方程[11]写出航天器的运动方程(将太阳帆给予的推力看作摄动量),将运动方程用轨道根数表示。轨道根数的几何意义见下图：

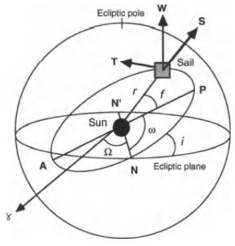

图 3: 轨道根数的几何意义[10]，其中S,W,T为太阳帆受到的三个方向上的力。

文中列举平面情形下需要使用的方程：

$$\begin{aligned} \dot{a} &= \frac{2}{n\sqrt{1-e^2}}[e\sin(f)f_r + (1+e\cos(f))f_t] \\ \frac{1}{2}\frac{de^2}{dt} &= \frac{e\sqrt{1-e^2}}{na}[\sin(f)f_r + (\cos(f)+\cos(E)f_t] \end{aligned} \tag{12}$$

其中$n = \sqrt{\frac{\mu}{a^3}}$, f_r和f_t分别代表径向和切向方向受到的力。通过轨道根数之间的关系，将方程组化简为：

$$\begin{aligned} \dot{a} &= \frac{2a^2}{h}[e\sin(f)f_r + \frac{p}{r}f_t] \\ \dot{e} &= \frac{1}{h}[p\sin(f)f_r + ((p+r)\cos(f)+er)f_t] \end{aligned} \tag{13}$$

将f_r, f_t替换为光压影响下的径向和切向加速度即可用于太阳帆的情形下，即：

$$\begin{aligned} f_r &= \frac{\beta\mu}{r^2}\cos^3(\alpha) \\ f_t &= \frac{\beta\mu}{r^2}\cos(\alpha)\sin(\alpha) \end{aligned} \tag{14}$$

若要表示远日点或近日点，可用半长轴和偏心率表达:

$$\begin{aligned} \dot{r_A} &= \dot{a}(1+e) + \dot{e}a \\ \dot{r_P} &= \dot{a}(1-e) + \dot{e}a \end{aligned} \tag{15}$$

局部最优化要求轨道的某一参数变化最快，即使该导数最大或最小，可以采用向量的形式以使这个问题更直白。现用η表示任意一个轨道根数。通过高斯卫星运动方程，高斯运动方程组的每个方程都可以整理为如下形式：

$$\dot{\eta} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} \cdot \begin{bmatrix} f_r \\ f_t \end{bmatrix} \tag{16}$$

因此，若要使导数最大/最小，需满足两向量共线，即$\frac{f_t}{f_r} = \frac{\lambda_2}{\lambda_1}$(相应地，如果要使变化最小，将两向量的点乘设为0即可)。已知太阳帆提供的力的两个分量的比值，即可求出力的方向α_f关于轨道根数的函数,其中α_f为力与径向方向的夹角:

$$\alpha_f = \tan^{-1}(\pm\frac{\lambda_2}{\lambda_1}) \tag{17}$$

正切函数内取正为增长最快，取负为减小最快。半长轴和偏心率变化最快的方法如下:

$$\begin{aligned} \tan(\alpha_f) &= \pm\frac{p}{re\sin(f)} \\ \tan(\alpha_f) &= \pm\frac{(p+r)\cos(f)+re}{p\sin(f)} \end{aligned} \tag{18}$$

显然,太阳帆所提供的力的方向并不能完全等于α_f，这是因为太阳帆获得的推力只能向"外", 而无法向"内"。为了求出最终需要的太阳帆朝向，还需将所得α_f代入公式(10)中的结论。

下面将列举五种局部最优控制策略，分别是偏心率增加、减小最快，半长轴增加最快，远、近日点增加最快。在图4到图8中，绘制其轨道，偏心率，半长轴，远日点随时间的变化。任务参数为轻度系数0.05，地球轨道出发，双曲剩余速度为3km/s，递推十五年。途中，蓝色圈代表航天器轨道与火星轨道交会，红色星号(*)代表该轨道上霍曼转移至木星轨道的最佳点。

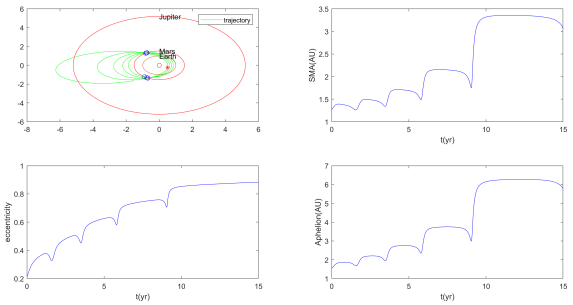

图 4: 偏心率增长最快策略仿真图

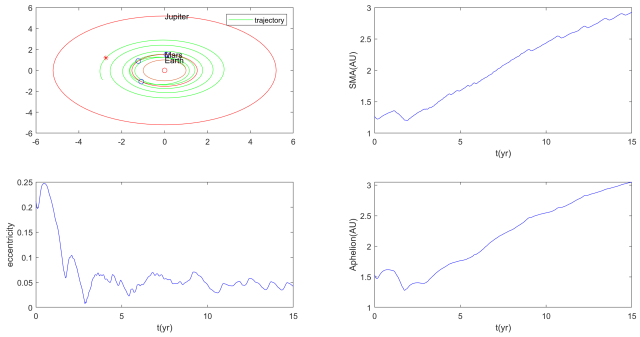

图 5: 偏心率减小最快策略仿真图

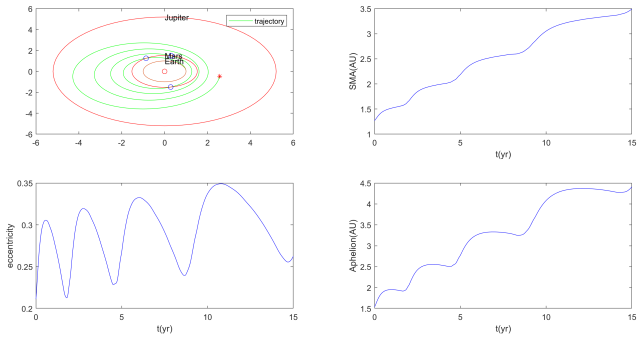

图 6: 半长轴增加最快策略仿真图

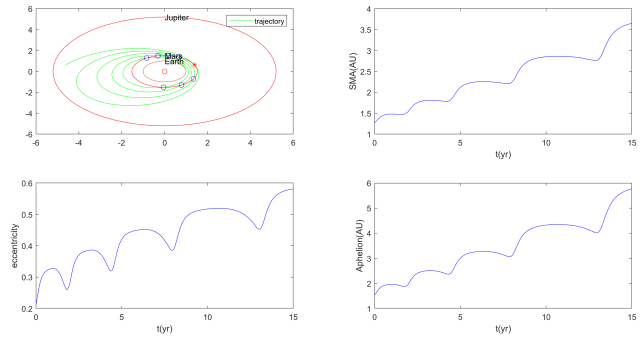

图 7: 远日点增加最快策略仿真图

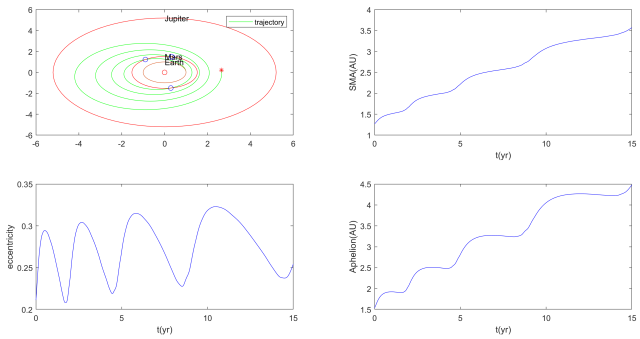

图 8: 近日点增加最快策略仿真图

从图4到图8中可以得出，偏心率增加的策略较为适合飞跃任务，但是由于偏心率大，飞跃行星时相对速度大，因此不适合此次任务。而最后的靠近阶段使用降低偏心率策略较为适合，使航天器切入时轨道偏心率尽可能小。这些策略将会在利用程序优化时进行具体筛选。

4.5 行星轨道飞行的力学模型

为了探究行星借力带来的影响，根据几何关系可获得借力前后各个量的关系。其中下标P代表行星相关的量:

$$\begin{aligned} v_\infty &= \sqrt{v_-^2 - v_P^2} \\ a &= -\frac{\mu_P}{v_\infty^2} \\ e &= 1 - \frac{r_P}{a} \\ \sin(\frac{\delta}{2}) &= \frac{1}{e} \\ \sin(\phi) &= \frac{v_-}{v_\infty}\sin(\gamma_-) \\ v_+ &= \sqrt{v_\infty^2 + v_P^2 - 2v_\infty v_P}\cos(\delta+\phi) \\ \sin(\gamma_+) &= \frac{v_\infty}{v_+}\sin(\delta+\phi) \end{aligned} \tag{19}$$

根据这些方程，利用几何关系求得飞越行星前后运动状态的变化。由于太阳帆推力小，因此向行星借力是优化轨道的一种重要手段。航天器在飞跃火星时速度变化由下图中几何关系体现:

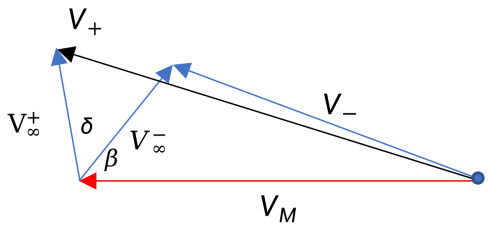

图 9: 行星借力（加速）

航天器进入木星轨道时，也用到相似的力学模型，此时求出入轨的速度增量。

由于目标的木星轨道半长轴确定，航天器入轨需要的速度变化量为:

$$\Delta V = \sqrt{\frac{2\mu_J}{r_P} + v_\infty^2} - \sqrt{\frac{2\mu_J}{r_P} - \frac{\mu_J}{a}} \tag{20}$$

其中r_P为近木点轨道半径。在半长轴与近木点确定的情况下，速度增量的大小仅与航天器的剩余速度v_∞相关，因此优化速度增量，即减小剩余速度。在后文中，为了与连

续小推力任务设计对比，也为了简洁起见，将航天器达到木星剩余速度为0km/s的状态时作为任务的终止条件。

5 模型求解

5.1 任务构思

对于局部最有策略进行有效的组合，即可实现近于最优化的结果[10]。有文献[5]提到可先增加半长轴，再降低偏心率使轨道成为圆形。这提供了一种思路，即先使用飞行较快的控制方法，再改为使用降低偏心率或其它使航天器轨道尽量靠近木星轨道的方法。

从第四部分中给出的结果来看，要使太阳帆航天器用同一种控制策略转移至木星，要么会显得过慢（如用减小偏心率的策略），要么则会有过大的径向速度，导致入轨机动速度增量过大，任务失败。因此需要采取一些折中方案。

为了弥补动力不足的情况，设计一次航天器向火星借力。由于航天器离开地球时没有额外的速度增量，可以通过借力获得一定的速度增量，使其转移能力增加。转移至火星要求并不是剩余速度小，而只是飞过，因此首先考虑使用两种最快的控制方式，即最快半长轴增加和最快远日点增加。

在航天器飞跃火星后，将其转移轨迹分作两段，即：可采用两种不同（或相同）的控制方法组合完成火星至木星的转移。最后航天器经过木星时给予一个速度增量，使得起相对于木星的剩余速度为0.

由于木星轨道半径大于5AU，因此太阳帆在航天器进入木星轨道后实际作用不大（实际上，很多任务会假设在5AU处抛弃太阳帆），不会使用太阳帆进行进一步变轨。为了减少入轨时的燃料消耗，此时设定在进入木星引力影响球时舍弃太阳帆。

5.2 程序设计

轨道设计的程序分为三大部分：计算程序，优化程序，主程序。

计算程序有积分器和飞跃检测程序。积分器程序为两部分，一部分为微分方程，使用ode45求解；另一部分为控制方法，使用一个对应的数调用（如输入1则采用最快半长轴增加的方法）。这两部分组成了积分器，执行最基础的计算。飞越检测程序会对需要检测的时间节点中每两个相邻节点进行检测，其运行过程如流程图10。

对于任务时间，地球-火星段固定，猜测其转移时间为3年，即设置最大积分时间为3年。对于火星-木星转移的两段，则设两段各自最大积分时间为15年。于是总任务时长被限制在33年内。

优化程序计算某一情况下，火星-木星转移轨道的控制方法，以及（如果中途更换控制方法）何时更换控制方法。ode45命令会返回一个离散的数组，因此优化程序在所有给出的值中迭代，其运行流程如图11。

最终由主程序控制其它程序，完成任务设计。

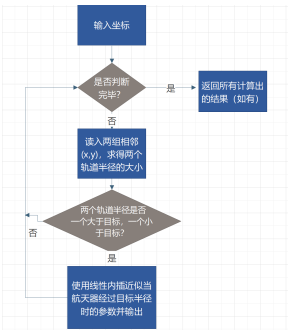

图 10: 飞越检测程序的流程图

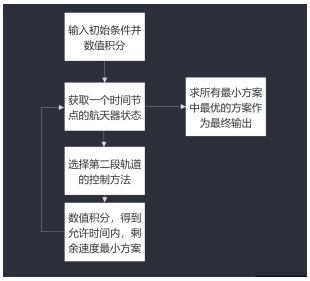

图 11: 优化程序的流程图

5.3 任务求解

任务求解方法为：设定转移阶段的第一阶段控制方法后，第二阶段最优的选择会在半长轴增加/减小，偏心率减小，近/远日点增加五种控制方法中挑选。通过更改第一阶段的控制方法，可以获得不同的最终结果。分别对于最快半长轴增加、最快偏心率增加、最快远日点半径增加、最快偏心率减小、最快近日点增加五种第一段策略进行讨论。在进行策略选取前，先对参数进行优化。

除了可以改变控制策略，也可以改变近火点以及太阳帆的轻度系数。首先对火星借力阶段进行讨论。已知航天器近火点高度越低，获得的速度增量越大，但应将近火点保持在安全距离。先将其设为100km。将半长轴和远日点控制方法进行比对，可发现前者经过火星时获得的加速度为3.0019km/s，后者则为3.4878km/s；同时，远日点增速最快的方案还有时间优势。显然，为了借力效果最好，应选择远日点增速最快的控制方案前往火星。再考虑近火点的选取。采用第一段轨道为半长轴增加策略的方案作为演示，分别求近火点为100km, 150km, 200km时的情况。

图中展示出，近火点高度越高，有效载荷就越小。这是因为近火点高度决定了飞行器借力获得的速度增量大小，而升高近火点则会减少速度增量。因此，近火点高度确定为100km。

航天器的轻度系数也是可优化的变量。轻度系数越大，太阳帆提供的推力越大，但

第一段策略	第二段策略	近火点高度（km）	第一段-第二段转换时刻	剩余速度（km/s）	有效载荷（kg）	转移总时长（年）
最快半长轴增加	最快近日点增加	100	12.19	1.61	3352	20.05
最快半长轴增加	最快半长轴增加	150	/	1.65	3319	20.08
最快半长轴增加	最快半长轴增加	200	/	1.67	3303	20.02

图 12: 有效载荷随近火点的变化表

是随之而来的就是太阳帆越重，因此需要权衡，使得有效载荷最大。下图展示了使用五种第一段策略进行优化的结果，其中对于每个策略也考虑了对轻度系数的优化。其中每一个策略的最优情况都在图中标出。

第一段策略	第二段策略	轻度系数	第一段-第二段转换时刻	剩余速度（km/s）	有效载荷（kg）	转移总时长（年）
最快半长轴增加	/	0.025	/	/	/	/
最快半长轴增加	最快近日点增加	0.05	12.19	1.61	3352	20.05
最快半长轴增加	最快偏心率减小	0.075	11.37	1.97	3031	18.66
最快半长轴增加	最快偏心率减小	0.1	5.07	0.25	4400	20.66
最快半长轴增加	最快偏心率减小	0.15	4.66	1.67	3071	10.36
最快半长轴增加	最快偏心率减小	0.125	0.27	1.89	2980	13.51
最快半长轴增加	最快偏心率减小	0.175	0.14	1.76	2949	11.15

第一段策略	第二段策略	轻度系数	第一段-第二段转换时刻	剩余速度（km/s）	有效载荷（kg）	转移总时长（年）
最快偏心率增加	/	0.025	/	/	/	/
最快偏心率增加	最快近日点增加	0.05	1.37	2.67	2637	17.6
最快偏心率增加	最快偏心率减小	0.075	0.41	3.1	2348	9.74
最快偏心率增加	最快偏心率减小	0.1	0.55	1.58	3257	16.12
最快偏心率增加	最快偏心率减小	0.125	1.23	1.2	3479	14.6
最快偏心率增加	最快偏心率减小	0.15	0.82	1.1	3491	13
最快偏心率增加	最快偏心率减小	0.175	2.74	0.92	3572	16.2
最快偏心率增加	最快偏心率减小	2	2.33	0.95	3475	15.29

第一段策略	第二段策略	轻度系数	第一段-第二段转换时刻	剩余速度（km/s）	有效载荷（kg）	转移总时长（年）
最快远日点增加	/	0.025	/	/	/	/
最快远日点增加	最快近日点增加	0.05	2.33	2.05	3030	19.05
最快远日点增加	最快近日点增加	0.075	0.27	2.83	2495	10.06
最快远日点增加	最快偏心率减小	0.1	2.6	1.04	3681	17.59
最快远日点增加	最快偏心率减小	0.125	12.6	0.88	3739	28.07
最快远日点增加	最快偏心率减小	0.15	3.29	0.33	4164	16.8
最快远日点增加	最快偏心率减小	0.175	1.1	0.65	3792	14.8

第一段策略	第二段策略	轻度系数	第一段-第二段转换时刻	剩余速度（km/s）	有效载荷（kg）	转移总时长（年）
最快偏心率减小	/	0.025	/	/	/	/
最快偏心率减小	最快偏心率减小	0.05	/	0.48	4333	31.16
最快偏心率减小	最快近日点增加	0.075	6.16	0.73	4015	22.31
最快偏心率减小	最快近日点增加	0.1	13.42	0.45	4208	28.3
最快偏心率减小	最快近日点增加	0.125	13.7	0.36	4212	22.73
最快偏心率减小	最快近日点增加	0.15	4.38	0.38	4116	13.08
最快偏心率减小	最快半长轴增加	0.175	6.85	0.59	3850	13.17

第一段策略	第二段策略	轻度系数	第一段-第二段转换时刻	剩余速度（km/s）	有效载荷（kg）	转移总时长（年）
最快近日点增加	/	0.025	/	/	/	/
最快近日点增加	最快近日点增加	0.05	12.88	1.98	3083	19.61
最快近日点增加	最快偏心率减小	0.075	10.96	1.88	3098	18.5
最快近日点增加	最快偏心率减小	0.1	1.64	0.56	4102	16.3
最快近日点增加	最快偏心率减小	0.125	8.22	1.6287	3159	18.18

图 13: 各个策略优化结果表

将图表中各个控制策略情况下的最优方案横向比对，发现轻度系数为0.1时，最快半长轴增加切换至最快偏心率减小的控制方案取得最优的有效载荷结果，约4400kg。使用主程序对任务设计进行绘图：

图中”To-Mars Tranfer”即地球-火星转移段，”Interplanetary Tranfer”即火星-木星转移的第一段，”Coasting Transfer”即航天器靠近木星时的转移段，也就是火星-木星转移的第二段。

从图中可见，航天器最终靠近木星轨道时，由于太阳帆轻度系数大，控制能力强，因此剩余速度小，以帆重的增加换取了化学燃料的减少。

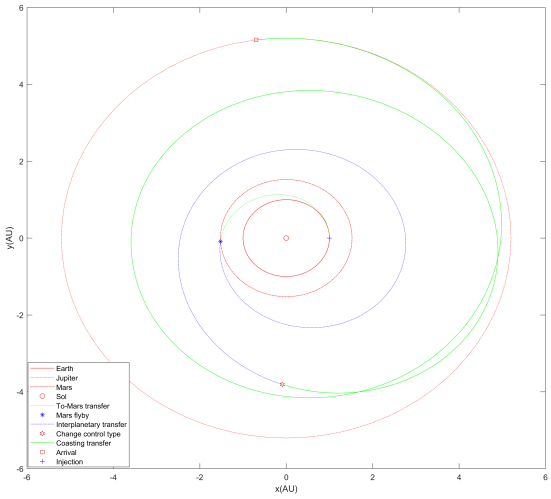

图 14: 地木转移的最优轨道仿真结果图

5.4 优化结果

通过上述的数值计算和分段优化，得出以下最优化策略与参数

地球逃逸速度增量: 0km/s,

初始质量: 5000kg,

抵达火星时间: 0.67yr,

近火点高度: 100km,

火星借力速度增量: 3.39km/s,

离开火星至切换控制策略时长: 5.07yr,

切换控制策略至抵达木星时长: 14.92yr,

到达木星剩余速度: 0.25km/s,

任务总时长: 20.66yr,

太阳帆质量: 344kg,

有效载荷: 4400kg,

转移序列: EMJ,

控制策略: 最快远日点增加，最快半长轴增加，最快偏心率减小.

与《行星际飞行轨道理论与应用》一书中脉冲推进的地木转移轨道设计相比，不论发射能量，还是抵达木星时的双曲线超速，使用太阳帆的设计具有绝对优势（书中所有

发射能量均大于10,且双曲线超速均大于5，EVEMJ序列还需要进行深空机动[6]，可见脉冲推进方式会消耗极多的燃料)。该书中另一种设计方法为离散脉冲设计方法生成的连续推力转移轨道设计，三个方案具体结果如图15，图中“剩余质量”指“航天器到达木星引力影响球”[6]时的质量，非入轨后质量；任务使用推力2N,比冲3500s的核电推进系统进行设计。

到达木星时间（yyyy/mm/dd）	2042/11/15	2043/12/28	2037/02/23
到达木星双曲线超速 $v_{\infty A}$ /（km·s^{-1}）	0.0	0.0	0.0
总飞行时间/年	6.878	8.0	6.658
总的速度增量/（km·s^{-1}）	11.476	13.294	14.016
探测器剩余质量/kg	3 385.480	3 182.674	3 105.544

图 15: 连续小推力地木转移轨道优化结果

可以看出，连续小推力的核电推进方式要优于脉冲推进，而太阳帆通过局部最优策略拼接的轨道，得出航天器到达木星引力影响球时的剩余质量远超其余两种传统推进方式，且仅经过火星借力，而非如书中设计的经过三次。虽然时间上，太阳帆转移时长为其它两种推进方式的三余倍，但由于其对于载荷有明显优势，不失为一种可取的策略。

6 任务总结

6.1 方法优势与成果

本文通过数值计算和分段建模的方法对使用太阳帆的地球-木星轨道进行了优化设计，阐明了这种方案在理论上的可行性以及优越性，获得了以下结论与成果：

（1）太阳帆足以胜任长期的行星际转移任务，且由于不消耗燃料，相较其它推进方式最大亮点在于可获得更大的有效载荷比，在本文的优化条件下达到了百分之88的有效载荷比；

（2）太阳帆的轻度系数和控制策略都是决定优化结果优劣的重要因素；

（3）由于使用了数值计算，使得计算相对简便快捷，可用于进行快速的轨道设计;

（4）程序模块化设计，可扩充更多转移段进行优化，便于进一步探究设计更复杂的任务;

（5）此地球-木星转移得到的结果可以进一步用于设计其它外围行星，如土星，的探测任务，具有参考意义。

6.2 方法不足与发展空间

（1）由于使用ode45积分器，在精度上有一定提高空间，在递推时间增加时会产生堆积误差，用于计算交会的程序使用线性内插，也会造成一定的误差；

（2）由于程序基于数值计算，在优化时得到的结果取决于步长。因此文中得出的方案仅为靠近最优方案，若要更精确则需增加步长；

（3）进行动力学方程建模时，忽略了摄动，行星遮罩，行星轨道偏心率、倾角，以及太阳帆老化的问题，在实际任务中这些都可能影响任务的完成;

（4）本文所得太阳帆任务转移时间远远大于其它策略，后续研究应着重在保证有效载荷的前提下，对时间进行优化。

综上所述，即便任务设计存在诸多不足之处，太阳帆所提供的行星际航行的可能性是非常多样化的，是一项值得持续研究发展的领域。本文中的方法给出了一种快速设计转移轨道的解决方案，且得出了比传统推进方式更优的有效载荷比，验证了此项技术在工程应用上的可行性与优越性。

7 致谢

作为一名关注航天的青年，在选择研究方面时注意到，继已在轨工作6余年的朱诺号木星轨道器后，全世界范围内并未再发射一颗木星轨道器（或是任何一颗探测外围行星的轨道器）。木星有着独特的大红斑，强烈的磁场，以及为地球扫清小行星的作用，对于研究太阳系的起源与发展等问题具有重要意义。随着朱诺号的老化，这位老将终将退出历史的舞台，坠落于木星大气层中。届时，木星轨道上可能再无人造航天器。这激发了我设计一个木星轨道器任务的想法。

太阳帆的想法则来自于与姚建总工程师的对话。他鼓励我进行“天马行空”的想象，设计一个突破传统方法局限的任务。我当即联想到了光帆二号用太阳帆进行卫星的轨道控制一例，决定研究太阳帆在行星际转移轨道上的应用。在阅读一些相关文献后，我发现前人所设计太阳帆轨道大多为对一整段进行全局优化，且大多为月球，周期性轨道，或火星任务，并未见远日行星轨道器的任务。另外，尽管不少著作会提到太阳帆的局部优化策略，甚至阐明可用其组合达到近于全局最优化的结果，但网络上并未见到此类成果，因此决定利用数值方法对这个问题进行探究。

设计程序的过程中也几经波折，调换过几次不同的策略，最终决定仅使用最基本的运动方程解决问题。程序计算的时间较长，是我之前未预料到的一点，幸而数值方法可以通过增加步长的方式进行调控和权衡，最终成功得出一个满意的结论。

文章的最后，我希望感谢我的父母，是他们的支持让我得以研究自己兴趣所在，让我在遇到困难时获得坚持的动力。感谢我的两位指导老师陈浦胤老师和李云飞老师，他们在任务的想法设计，调试，和呈现方面提供了宝贵的指导。另外，感谢上海宇航系统工程研究所的李绿萍老师，虽然不是主要指导老师，她对于任务的设计和结果呈现也提供了大量的意见。最终，一并感谢其他帮助过我的导师，朋友。

8 附录：程序

所有程序均为自行编写，在Matlab R2019b环境下运行。

8.1 主程序

```
clear;
%constants
b = 0.05;
rE = 149.598e6;
rM = 227.956e6;
AU = rE;
rJ = 778.279e6;
vE = 29.7846;
vJ = 13.06;
m = 132.712e9;
PerHeight = 100;
Type_Coast = 3; %rise aphelion
Type_pass = 4;
%initialize plot%
hold off;
APHA=0:pi/40:2*pi;
Circx=cos(APHA);
Circy=sin(APHA);
plot(Circx,Circy,'r-'); hold on;
Circx=778279000*cos(APHA)/AU;
Circy=778279000*sin(APHA)/AU;
plot(Circx,Circy,'r:'); hold on;
Circx = 227956000*cos(APHA)/AU;
Circy=227956000*sin(APHA)/AU;
plot(Circx, Circy, 'r-.'); hold on;
plot(0, 0,'ro'); hold on;
xlabel('x(AU)');
ylabel('y(AU)');
%###-coast to mars orbit-###%

Tspan_Coast = [0:86400:86400*365*3];
Tspan_Prop = [0:86400:86400*365*15];
pars_Coast = [b, Type_Coast];
```

```
%---coasting propagation---%
InitSet = [rE,0, 0, vE];
disp("###Mars Transfer###");
disp("Propagating to Mars with initial state... ");
disp(rE+"km; "+0+"km; "+0+"km/s; "+ vE+"km/s");
TargetMars = stopAtRadius(InitSet(1), InitSet(2), rM, InitSet(3),..
 InitSet(4), b, Type_Coast, 3*365);
if isempty(TargetMars) == 1
    disp("Transfer error, check initialization");
else
    disp("-Transfer correct-");
    disp("Flyby at t = "+double(TargetMars(5)/(86400*365))+" (yr)");
end
FirstPass = TargetMars(:,1);
[t_mars,out_mars] = ode45(@(t,pos) EulerIntegrateTest(t,...
 pos, pars_Coast), Tspan_Coast, InitSet);
%-delete redundant data-%
t_mars = t_mars(1:FirstPass(6)+2);
out_mars = out_mars(1:FirstPass(6)+2,:);
%plot first segment%
plot(out_mars(:,1)./AU,out_mars(:,2)./AU,'g:'); hold on;
plot(FirstPass(1)/AU, FirstPass(2)/AU, 'b*'); hold on;

%---Renew for Martian pass---%
disp("###Mars Flyby###");
AfterPass = MarsPassing(FirstPass(1:4),PerHeight);
disp("Velocity Increase: "+double(AfterPass(6))+"km/s");
disp("Periapsis height: "+double(PerHeight)+"km");
%---Mars-Jupiter transfer segment---%
PassSet = [FirstPass(1),FirstPass(2), AfterPass(1), AfterPass(2)];
pars_pass = [b,Type_pass];
[t_pass,out_pass] = ode45(@(t,pos) EulerIntegrateTest(t,...
 pos, double(pars_pass)), Tspan_Prop,double(PassSet));
%disp results
disp("###Interplanetary Transfer###");
disp("Injection State...");
disp(double(FirstPass(1))+"km; "+double(FirstPass(2))+"km; "+...
double(AfterPass(1))+"km/s; "+double(AfterPass(2))+"km/s");
%---analyze commands---%
```

```
BestBreaks = [];
for j = 1:108
    CurrentState = out_pass(j*50,:);
    BestNow = ControlFinalOpt(CurrentState);
    if length(BestNow) ~= 3
        BestNow = [NaN, NaN, NaN];
    end
    BestBreaks = [BestBreaks,BestNow'];
end

%-find best value in BestBreaks-
BestBreaks = double(BestBreaks);
dVs = BestBreaks(1,:);
mindVglobal = min(dVs);
minIndex = find(dVs == mindVglobal);
%disp(BestBreaks(1,minIndex));
BestChoice = double(BestBreaks(:,minIndex));
%-plot answer-%
t_pass= t_pass(1:minIndex*50);
out_pass = out_pass(1:minIndex*50,:);
plot(out_pass(:,1)./AU,out_pass(:,2)./AU,'b:'); hold on;
plot(out_pass(minIndex*50,1)/AU, out_pass(minIndex*50,2)/AU,'rh'); hold on;
disp("-Transfer correct-")
disp("Control type change at t = "+double(minIndex*50/365)+"yr");
%plot final segment
disp("###Coasting Transfer###")

final_init = out_pass(minIndex*50,:);
typesArray = [1,3,4,6,7,8];
disp("Injection State...");
disp(double(final_init(1))+"km; "+double(final_init(2))+"km; "+...
double(final_init(3))+"km/s; "+double(final_init(4))+"km/s");
Type_final = typesArray(BestBreaks(2,minIndex));
[t_arr,out_arr] = ode45(@(t,pos) EulerIntegrateTest(t,..
 pos, [b,Type_final]),[1:86400:86400*365*15],final_init);
optFinal = stopAtRadius(final_init(1), final_init(2), rJ,...
 final_init(3), final_init(4),b, Type_final, 15*365);
Vrs =(optFinal(1,:).*optFinal(3,:)+optFinal(2,:).*optFinal(4,:))./rJ;
Vts =  ((optFinal(3,:).^2+optFinal(4,:).^2)-Vrs.^2).^0.5;
```

```
dVs = (Vrs.^2+(Vts-vJ).^2).^0.5;
minfindV  = min(dVs);
finIndex = find(dVs == minfindV);
finalChoice = optFinal(:,finIndex);
%disp results
disp("-Transfer correct-");
disp("Arrival at t = "+double(finalChoice(5)/(365*86400))+" (yr)");
disp("Arrival V-infinity = "+double(minfindV)+" (km/s)");
%-clear data and plot-%
out_arr = out_arr(1:finalChoice(6)+2,:);
plot(out_arr(:,1)./AU,out_arr(:,2)./AU,'g-'); hold on;
plot(finalChoice(1)/AU,finalChoice(2)/AU,'rs'); hold on;
plot(rE/AU,0/AU,'b+');
legend('Earth','Jupiter','Mars','Sol','To-Mars transfer','Mars flyby',...
'Interplanetary transfer','Change control type','Coasting transfer','Arrival','Injection');

%disp spec results
disp("###Spacecraft Parameters###");
disp("Lightness Factor: "+b);
disp("Specific Impulse: 450 seconds");
disp("Sail density: 1.0g/m^2");
SigIdl = 1.53*(1.9)/2000;
Sig = SigIdl/b;
Ms = 0.0010*5000/Sig;
disp("Sail Mass: "+Ms+"kg");
Mr = (5000-Ms)/(exp(minfindV/(450*0.0098)));
disp("Effective Payload: "+double(Mr)+"kg");
disp("------");
disp("Solution converged");
disp("Total Transfer duration: "+double(double(TargetMars(5)/(86400*365))+...
double(finalChoice(5)/(365*86400))+double(minIndex*50/365))+" (yr)");
disp("Transfer Type: EMJ/"+Type_Coast+""+Type_pass+""+Type_final);
```

8.2 优化程序

```
function allBest = ControlFinalOpt(flyPos)
b = 0.05;
typesArray = [1,3,4,6,7.8];
CollectReturn = []; %initialize
```

```
CollectTime = [];
for i = 1:length(typesArray)
    CurrType = typesArray(i);
    returnOpt = FinalTransferOptimize([b, CurrType],double(flyPos));
    CollectReturn = [CollectReturn, returnOpt(1)];
    CollectTime = [CollectTime, returnOpt(2)];
end
bestdV = min(CollectReturn);
index = find(CollectReturn==bestdV);
bestTime = CollectTime(index);
%disp(index);
allBest=[bestdV,index,bestTime];
end
```

8.3 飞越检测程序

```
function [STAT] = stopAtRadius(Initialx, Initialy, Target,InitVx, InitVy, b, type, MaxTime)
    %this program helps to stop the spacecraft
    %when spacecraft orbital radius exceed target
    %STAT = []; %initialize
    MaxSearch = MaxTime*86400; %maxtime is in days!
    if MaxSearch<86400*365
        %disp("Search time is too small!");
    end
    %-Calculate Trajectory until time of MaxSearch-%
    Pos0 = [Initialx, Initialy, InitVx, InitVy];
    pars = [b, type];
    [t,out] = ode45(@(t,pos) EulerIntegrateTest(t, pos, pars), [0:86400:MaxSearch],Pos0);
    %initialize storage
    finds=[]; %format: t-first, t-cross, arrange by column
    %now check for all instances that resemble a crossing
    r = (out(:,1).^2+out(:,2).^2).^0.5;
    for i=1:MaxTime-1
        if r(i)<Target
            if r(i+1)>Target
               %disp(["Report crossing at timestep" ,t(i),"to" , t(i+1)]);
                %-record values-%
                finds=[finds, [(i), (i+1)]'];
            end
```

```
        end
        if r(i)>Target
            if r(i+1)<Target
                %disp(["Report crossing at timestep ",t(i), "to" , t(i+1)]);
                finds=[finds, [(i), (i+1)]'];
            end
        end
    end
    %analyze crossing events
    if size(finds, 2) == 0
        %disp("No Cross found");
        STAT = [];
    else
        State = [];
        for j=1:size(finds,2)
            %--prior to cross--%
            tp = t(finds(1,j));
            xp = out(finds(1,j),1);
            yp = out(finds(1,j),2);
            dxp = out(finds(1,j),3);
            dyp = out(finds(1,j),4);
            %disp(dxp)
            %--after cross--%
            ta = t(finds(2,j));
            xa = out(finds(2,j),1);
            ya = out(finds(2,j),2);
            dxa = out(finds(2,j),3);
            dya = out(finds(2,j),4);
            clear X; clear Y;
            syms X Y;
            if xa>xp
                assume(X>xp); assume(X<xa);
            else
                assume(X>xa); assume(X<xp);
            end

            eq1 = X^2+Y^2-Target^2;
            eq1 = subs(eq1);
            eq2 = (yp-ya)*X+(xa-xp)*Y+xp*ya-xa*yp;
```

```
            eq2 = subs(eq2);
            [X,Y] = solve(eq1, eq2);
            if abs(X(1)-xp)+abs(X(1)-xa) == abs(xa-xp)
                rx = X(1);
                ry = Y(1);
            else
                rx = X(2); ry = Y(2);
            end
            rt = tp+(ta-tp)*sqrt((rx-xp)^2+(ry-yp)^2)/sqrt((xa-xp)^2+(ya-yp)^2);
            rdx = dxp+sqrt((rx-xp)^2+(ry-yp)^2)/sqrt((xa-xp)^2+(ya-yp)^2)*(dxa-dxp);
            rdy = dyp+sqrt((rx-xp)^2+(ry-yp)^2)/sqrt((xa-xp)^2+(ya-yp)^2)*(dya-dyp);
            if j == 1
                STAT =  [rx,ry,rdx,rdy,rt,tp/(ta-tp)]';
            else
                STAT = [STAT, [rx,ry,rdx,rdy,rt,tp/(ta-tp)]'];
            end
        end
    end
end
```

8.4 积分器

```
function Elems = EulerIntegrateTest(t,pos,pars)
    %Position Configs: x, y, dx, dy
    b = pars(1);
    type = pars(2);
    x = pos(1);
    y = pos(2);
    r = sqrt(pos(1)^2+pos(2)^2);
    if  x>0 && y>=0
        thta = atan(pos(2)/pos(1));
    elseif x<0 && y >=0
        thta = atan(pos(2)/pos(1))+pi;
    elseif x<0 && y<=0
        thta = atan(pos(2)/pos(1))+pi;
    elseif x>0 && y<=0
        thta = atan(pos(2)/pos(1))+2*pi;
    else
        if y>0
```

```
            thta = pi/2;
        else
            thta = 3*pi/2;
        end
    end
    %define orbital components
    m = 132.712e9; %solar grav
    h = pos(1)*pos(4)-pos(2)*pos(3); %angula momentum of unit mass
    p = h^2 /m; %semi lactus
    a = (2/r-(pos(3)^2+pos(4)^2)/m)^(-1); %SMA
    e = sqrt(1-p/a);
    %disp([thta,e]);
    %Setup solar sail force orientation
    %This section uses type to determine which control method is used
...
...
...
    %Values of type: SMA+,ECC+,APO+ ECC-, CONST,PER+,ECC Keep, SMA-
    %--forces--%
%-Gravitation-%
g = m/r^3;
gx = -g*pos(1);
    gy = -g*pos(2);
%-SRP-%
srp_r = b*m*cos(apha)^2/r^3 *cos(apha);
     srp_t = b*m*cos(apha)^2/r^3 *sin(apha);
sx = srp_r*pos(1)-srp_t*pos(2);
sy = srp_r*pos(2)+srp_t*pos(1);
%accel components
ax = gx+sx;
ay = gy+sy;
    %accel in vector form
    Elems = [real(pos(3)); real(pos(4)); real(ax); real(ay)];
end
```

8.5 借力计算程序

```
function Pass = MarsPassing(inState,PerHeight)
```

```
%---this program analyzes the gravitation slingshot for mars passing,
%or indeed any other planet with a change in the gravitation constant.
m = 42828; %grav const
body = 3389.5;
vM = 24.07;
% get Vr and Vt
inx = inState(1);
iny = inState(2);
indx = inState(3);
indy = inState(4);
inPos = [inx, iny];
inVel = [indx, indy];
inradial = inPos./sqrt(dot(inPos,inPos));
invr = dot(inVel, inradial);
invt = sqrt(indx^2+indy^2-invr^2);
ingma = atan(invr/invt); %incidence path angle
%find polar angle
if  inx>0 && iny>=0
    thta = atan(iny/inx);
elseif inx<0 && iny >=0
    thta = atan(iny/inx)+pi;
elseif inx<0 && iny<=0
    thta = atan(iny/inx)+pi;
elseif inx>0 && iny<=0
    thta = atan(iny/inx)+2*pi;
else
    if iny>0
        thta = pi/2;
    else
        thta = 3*pi/2;
    end
end
%-now transfer into planetary coord-%
vinf = sqrt(invr^2+(invt-vM)^2); %hyperbolic excess velocity
a = -m/vinf^2; %approximate for SMA
rp = PerHeight+body; %set closest approach
e = 1-rp/a;
delta = 2*asin(1/e); %find deflect angle
beta = asin(sin(ingma)*sqrt(indx^2+indy^2)/vinf);
```

```
    %-find exit velocity-
    outv = sqrt(vinf^2+vM^2-2*vinf*vM*cos(beta+delta));
    outgma = asin((vinf/outv)*sin(beta+delta));
    changeV = 2*vinf/(1+(rp*vinf^2/m)); %linear velocity gained through maneuver
    %-change into radial and transverse elements
    outvr = outv*sin(outgma);
    outvt = outv*cos(outgma);
    %-change into cartesian elements;
    outvx = outvr*cos(thta)-outvt*sin(thta);
    outvy = outvr*sin(thta)+cos(thta)*outvt;
    %expot elements
    Pass = [outvx, outvy,outvr,outvt, outgma, changeV];
end
```

参考文献

[1] W.A.Hollerman, The Physics of Solar Sails, Nasa Faculty Membership Program, 2002.

[2] Yufei Guo, Dongzhu Feng, Xin Wang, Cong Li, Yunzhao Liu, "The Earth-Mars Transfer Trajectory Optimization of Solar Sail Based on hp-Adaptive Pseudospectral Method", Discrete Dynamics in Nature and Society, vol. 2018, Article ID 6916848, 14 pages, 2018. https://doi.org/10.1155/2018/6916848

[3] https://dspace.mit.edu/bitstream/handle/1721.1/37176/27479802-MIT.pdf?sequence=2

[4] Zeng, Xiangyuan and Vulpetti, Giovanni and Circi, Christian. (2018). Solar sail H-reversal trajectory: A review of its advances and applications. Astrodynamics. 3. 1-15. 10.1007/s42064-018-0032-y

[5] Dachwald, Bernd and Ohndorf, Andreas and Wie, Bong. (2006). Solar Sail Trajectory Optimization for the Solar Polar Imager (SPI) Mission. 10.2514/6.2006-6177.

[6] 尚海滨，行星际飞行轨道理论与应用，北京理工大学出版社，2019.

[7] R.Miller, Solar Sails - The math behind solar sails, 2015.

[8] http://www2.ee.ic.ac.uk/derek.low08/yr2proj/solarsails.htm

[9] Richard Rieber, Principals of Solar Sailing

[10] C.R.McInnes, Solar Sailing: Technology, Dynamics, and Mission Applications, Springer, 1999.

[11] 蒋春华, 徐天河, 乔晶, 等. 拉格朗日/高斯无奇点卫星运动方程推导与分析[J]. 测绘学报，2018，47(4)：455-464. DOI: 10.11947/j.AGCS.2018.20170082

[12] Bernd Dachwald and Wolfgang Seboldt, Potential Solar Sail Degradation Effects on Trajectory and Attitude Control