2014 高教社杯全国大学生数学建模竞赛

承　诺　书

我们仔细阅读了《全国大学生数学建模竞赛章程》和《全国大学生数学建模竞赛参赛规则》（以下简称为“竞赛章程和参赛规则”，可从全国大学生数学建模竞赛网站下载）。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛章程和参赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛章程和参赛规则，以保证竞赛的公正、公平性。如有违反竞赛章程和参赛规则的行为，我们将受到严肃处理。

我们授权全国大学生数学建模竞赛组委会，可将我们的论文以任何形式进行公开展示（包括进行网上公示，在书籍、期刊和其他媒体进行正式或非正式发表等）。

我们参赛选择的题号是（从 A/B/C/D 中选择一项填写）：A

我们的报名参赛队号为（8 位数字组成的编号）：10017013

所属学校（请填写完整的全名）：江南大学

参赛队员 (打印并签名) ：1. 高艺哲

2. 胡茂承

3. 吴奕

指导教师或指导教师组负责人 (打印并签名)：魏国强

（论文纸质版与电子版中的以上信息必须一致，只是电子版中无需签名。以上内容请仔细核对，提交后将不再允许做任何修改。如填写错误，论文可能被取消评奖资格。）

日期：2014 年 9 月 15 日

赛区评阅编号（由赛区组委会评阅前进行编号）：

2014 高教社杯全国大学生数学建模竞赛

编 号 专 用 页

赛区评阅编号（由赛区组委会评阅前进行编号）：

赛区评阅记录（可供赛区评阅时使用）：

全国统一编号（由赛区组委会送交全国前编号）：

全国评阅编号（由全国组委会评阅前进行编号）：

嫦娥三号软着陆轨道设计与控制策略

摘要

本文以嫦娥三号飞行的制导与软着陆控制问题为背景，建立动力模型，利用迭代算法、螺旋搜索算法，求解六个状态的最优控制策略和着陆轨道。

针对问题一，要求出近、远月点位置，以及卫星相应点的速度。以经、纬、高为坐标建立三维月心坐标系。假设主减速终止位置在着陆点上方，根据运动学方程、差分方程，利用MATLAB求解出近月点与着陆点两点坐标的极角之差$\alpha \approx 12.83^{o}$，再由着陆点位置推导出近月点的位置（$19.51^{o}W,31.29^{o}N,15km$），远月点的位置（$160.49^{o}E,31.29^{o}S,100km$），两点速度$v_{近月}$=$1.6922km/s$，$v_{远月}$=$1.6139km/s$，方向分别为轨道切向方向。

针对问题二，要确定卫星的着陆轨道以及六个状态的最优控制策略。将着陆轨道主要分为减速过程与避障过程。主减速过程以燃料消耗量最小为优化目标，将轨道曲线用多段折线近似表示，建立动力模型，利用迭代算法逐步求解飞行器运动状态。主减速过程燃料消耗质量1096.9kg,飞行时间430s，水平位移390.66km，推动力7500N。快速调整阶段主要是将卫星水平速度降为0，飞行时间7.9s。粗避障阶段利用螺旋搜索模型，基于离散二维熵的最优阈值选择模型，粗步避开大陨石坑，飞行时间50.94s。精避障阶段采用同样方法，精细避开月面障碍物，飞行时间9.35s。其轨道方程以及最优控制策略见问题二分析与解答。

针对问题三，将本模型计算出的数据与文献记录的实际数据比较，针对模型建立过程中忽略的变量以及简化过程对结果产生的影响进行误差分析。根据变量对燃料消耗质量的影响程度进行敏感性分析，变量比冲，推动力，操纵角的敏感程度分别为0.154%，1.471%，0.387%。

关键词：动力模型　迭代算法　螺旋搜索算法　离散二维熵

1　问题的重述

问题的背景：嫦娥三号于 2013 年 12 月 2 日 1 时 30 分成功发射，12 月 6 日抵达月球轨道。嫦娥三号在着陆准备轨道上的运行质量为 2.4t，其安装在下部的主减速发动机能够产生 1500N 到 7500N 的可调节推力，其比冲（即单位质量的推进剂产生的推力）为 2940m/s，可以满足调整速度的控制要求。在四周安装有姿态调整发动机，在给定主减速发动机的推力方向后，能够自动通过多个发动机的脉冲组合实现各种姿态的调整控制。嫦娥三号的预定着陆点为 19.51W,44.12N,海拔为-2641m。

问题的要求：嫦娥三号在高速飞行的情况下，要保证准确地在月球预定区域内实现软着陆，关键问题是着陆轨道与控制策略的设计。其着陆轨道设计的基本要求：着陆准备轨道为近月点 15km，远月点 100km 的椭圆形轨道；着陆轨道为从近月点至着陆点，其软着陆过程共分为 6 个阶段，要求满足每个阶段在关键点所处的状态；尽量减少软着陆过程的燃料消耗。

根据上述的基本要求，需要建立数学模型解决的问题：

（1）确定着陆准备轨道近月点和远月点的位置，以及嫦娥三号相应速度的大小与方向。

（2）确定嫦娥三号的着陆轨道和在 6 个阶段的最优控制策略。

（3）对于你们设计的着陆轨道和控制策略做相应的误差分析和敏感性分析。

2　问题的分析

题目的第一问可以分解成两个小问。问题（1）要求解嫦娥三号在近月点和远月点的速度。假设月球质心是在椭圆轨道的焦点上，这样椭圆轨道的长短轴和离心率是已知的，同时月球质量是已知的，这样直接根据天体物理学方程，可以在不考虑近、远月点具体位置的情况下直接求出嫦娥三号在近、远月点的速度大小，方向为轨道切线方向。问题（2）要确定着陆准备轨道近月点和远月点的位置。这是一个有约束非线性的优化问题，优化目标是使燃料消耗量最小。根据预着陆点的位置以及主减速阶段水平位移大小确定近月点位置。可以对飞行器的减速过程建立动力学模型。为了简化模型，嫦娥三号距月 15km 到 3km 的主减速过程，假设卫星在离地 3km 处正好位于预着陆点正上方，同时将推动力大小视为定常力，方向与飞行器运动方向相反。这样处理后该模型的约束量是推动力方向的调节范围，飞行器垂直

方向的下落距离。简化动力学模型，求出卫星主减速过程的水平位移，结合预着陆点的位置推出近月点位置，再根据天体知识，推导出远月点位置。

题目的第二问是根据尽量减少软着陆过程的燃料消耗这一目标，来确定卫星的着陆轨道以及六个状态的最优控制策略。而软着陆过程可主要分为减速运动与避障过程，其中，主减速过程已在问题一中得到解决。减速过程可根据起始点位置，速度，加速度，推力，操纵角β等变量之间关系式以及运动约束条件，在卫星质量减少最小的条件下，求解相应的轨道。卫星在离地较近处基本位于目标上方，月球表面的平整度无法改变，则避障过程中以避障为最高级的优化目标。上一个状态的轨道会影响下一状态的飞行方式，为了简化模型，根据初始值或上一状态的终止值依次求解六个状态的每一状态的最优控制。

针对问题三，敏感性分析是当推动力，操纵角，比冲等变量发生变化时，观察燃料消耗质量的变化。误差分析是分析忽略掉的变量以及简化过程和系统误差在建模时造成目标函数值与准确值的差异。

3　符号约定

a ：着陆准备轨道（椭圆）的半长轴。

b ：着陆准备轨道（椭圆）的半短轴。

c ：着陆准备轨道（椭圆）的半焦距。

T ：卫星在着陆准备轨道运动的周期。

e ：椭圆的离心率。

ρ ：曲率半径。

v_1 ：卫星在近月点的速度大小。

v_2 ：卫星在远月点的速度大小。

r_1 ：近月点距月心的距离。

r_2 ：远月点距月心的距离。

v_r ：径向速度。

v_θ ：椭圆轨道中横向速度。

r ：某点距月心的距离。

$\dot{r}$ ：某点距月心的距离对时间的导数。

θ ：图 2 中设定的夹角。

$\dot{\theta}$ ：夹角θ对时间的导数。

M ：月球质量，题给 $M = 7.3477 \times 10^{22} kg$ 。

G ：万有引力常量，$G = 6.67 \times 10^{-11} N \cdot m^2 \cdot kg^{-2}$ 。

v_α ：椭圆轨道中横向速度。

a ：加速度。

T ：推力大小。

β ：推力与当地水平线的夹角。

m ：嫦娥三号的瞬时质量。

I_{sq} ：比冲，题给值 $I_{sq} = 2940 m / s$ 。

H ：离散二维熵。

4 模型假设

假设一：嫦娥三号软着陆过程中不受月球引力非球项、日月引力摄动等影响。

假设二：嫦娥三号在主减速过程中推动力大小恒定。

假设三：月球是标准的圆球，月球球心在椭圆轨道的焦点上。

假设四：着陆过程不考虑月球自转的影响。

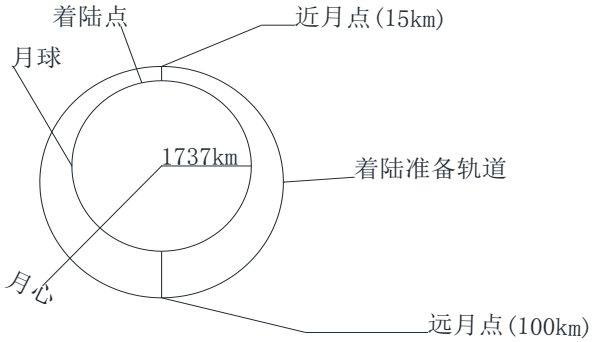
假设五：在避障过程中一定存在安全着陆点。

5 模型的建立与求解

5.1 问题一

5.1.1 问题（1）

求解卫星在近月点与远月点的速度大小是一个天体物理学问题[1]，



图一 卫星环月轨道示意图

观察图一，月球平均半径为1737.0km，可根据椭圆方程求出椭圆轨道的长轴a=1794.5km，短轴b=1794.0km。

行星经过近日点和远日点时，由于椭圆具有对称性，行星运动的曲率半径相同。那么，设该点在此椭圆轨道的曲率半径是ρ ，对于经过近日与远日点时分别有：

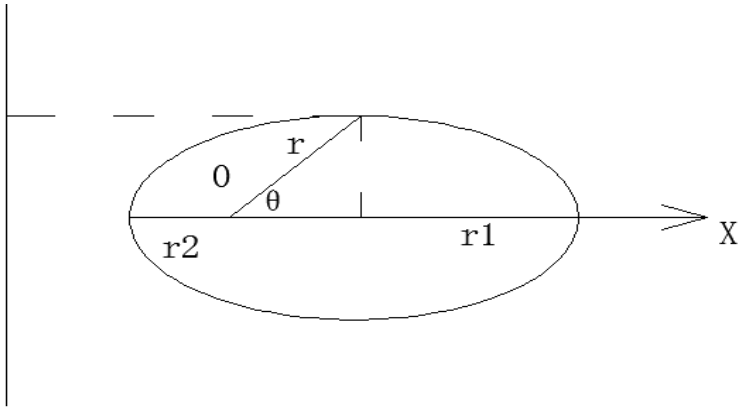
$$a_1 = \frac{v_1^2}{\rho}, a_2 = \frac{v_2^2}{\rho} \tag{1}$$

由(1)可得：

$$\frac{v_1^2}{v_2^2} = \frac{a_1}{a_2} = \frac{\frac{GM}{r_1^2}}{\frac{GM}{r_2^2}} = \frac{r_2^2}{r_1^2} \tag{2}$$

对（3）式两边同时消方，得：

$$\frac{v_1}{v_2} = \frac{r_1}{r_2} \tag{3}$$



图二 卫星椭圆轨道几何示意图

如图二，在极坐标中，对于径向速度和横向速度有：

$$v_r = \dot{r}, v_\theta = r\dot{\theta} \tag{4}$$

由椭圆方程方程

$$r = \frac{e\rho}{1 - e\cos\theta} \tag{5}$$

由（5）推导得：

$$\frac{e\rho}{r} = 1 - e\cos\theta \tag{6}$$

对（6）式两边对时间求导，有：

$$-\frac{e\rho\dot{r}}{r^2} = e\sin\theta\dot{\theta} \tag{7}$$

再对（7）式整理可得：

$$\dot{r} = -\frac{r^2}{\rho}\sin\theta\dot{\theta} \tag{8}$$

行星运动的速度为：

$$v = \sqrt{v_r^2 + v_\theta^2} = \sqrt{\dot{r}^2 + \left(r\dot{\theta}\right)^2} = \frac{r^2\dot{\theta}}{e\rho}\sqrt{1 - 2e\cos\theta + e^2} \tag{9}$$

综上，

$$v = \frac{r^2\dot{\theta}}{e\rho}\sqrt{1 - 2e\cos\theta + e^2} \tag{10}$$

上式中的$r^2\dot{\theta}$表示单位时间内矢径扫过的面积的 2 倍，因此有

$$r^2\dot{\theta} = \frac{2\pi ab}{T} \tag{11}$$

其中 a 表示椭圆的半长轴，b 表示椭圆轨道的半短轴，T 表示卫星运动的周期。

由“椭圆轨道上的卫星的速度和能量”中可知

$$a = \frac{e\rho}{1 - e^2} \tag{12}$$

$$b = a\sqrt{1 - e^2} \tag{13}$$

由椭圆方程可得

$$e\cos\theta = 1 - \frac{e\rho}{r} \tag{14}$$

根据开普勒第三定律，有：

$$\frac{a^3}{T^2} = k \tag{15}$$

其中，(15)式中的 k 为常数。

将(11)式至（15）式分别代入（10）式，得：

$$v = \frac{2\pi ab}{T} \cdot \frac{1}{e\rho}\sqrt{\frac{2e\rho}{r} + e^2 - 1} = b\sqrt{\frac{4\pi^2 k}{a}\left(\frac{2}{re\rho} + \frac{e^2-1}{e^2\rho^2}\right)} \tag{16}$$

令 $4\pi^2 k = GM$ ，其中，M 表示中心天体月球的质量，继续整理（16）式，

$$v = b\sqrt{\frac{GM}{a}\left(\frac{2}{re\rho} + \frac{e^2-1}{e^2\rho^2}\right)} = \sqrt{\frac{GM}{a}} \cdot \sqrt{\frac{2a}{r} - 1} \tag{17}$$

即：

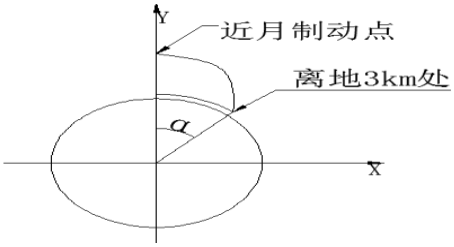
$$v = \sqrt{\frac{GM}{a}} \cdot \sqrt{\frac{2a}{r} - 1} \tag{18}$$

将 $M = 7.3477 \times 10^{22} kg$， $G = 6.67 \times 10^{-11} N \cdot m^2 \cdot kg^{-2}$， $a = 1794.5km$ ，$r_1 = 1752.013km$ ， $r_2 = 1837.013km$ 代入（18）式，

计算得嫦娥三号在近月点的速度为 1.6922km/s,远月点的速度为 1.6139km/s。

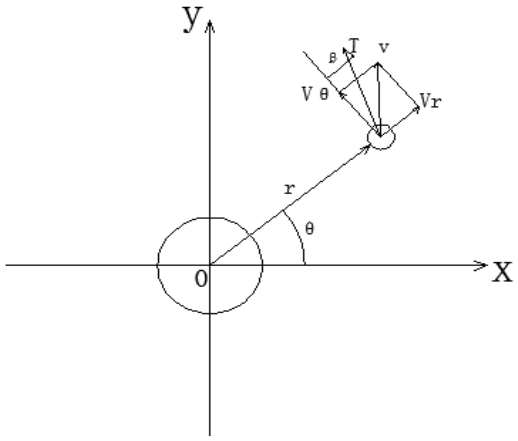
5.1.2 问题（2）

求解卫星的近月点与远月点的位置是一个有约束非线性规划问题。



图三：近月点与离地 3km 之间的曲线示意图

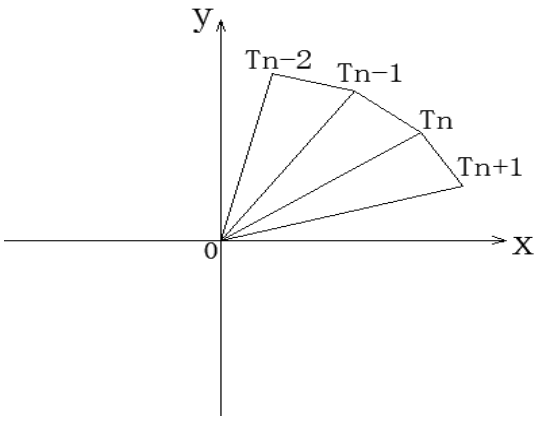
如图三，建立月心二维坐标系:原点为月心，Y 轴由由月心指向软着陆制动起始点即近月点，建立的 X 轴使得近月点到离地 3km 之间的曲线落在 X、Y 轴构成的平面内。



图四：飞行器受力分析示意图

如图五所示，在惯性坐标系中，以月心为原点的极坐标形式受控飞行器动力学方程[2] 为：

$$\begin{cases} \dot{V}_r = -\dfrac{\mu}{r^2} + \dfrac{V_\alpha^2}{r} + a\sin\beta \\ \dot{V}_\alpha = -\dfrac{V_r V_\alpha}{2} + a\cos\beta \\ \dot{r} = V_r \\ \dot{\alpha} = \dfrac{V_\alpha}{r} \end{cases} \tag{19}$$



图五：主减速运动曲线分解示意图

将主减速阶段的类抛物线分解为多段折线。当折线段数足够大时，可以视为在每一阶段上，飞行器的飞行过程近似为匀加速直线运动，并且飞行器因燃料消耗产生的质量变化是可以忽略不计的。在每一阶段的终点，飞行器的状态是下一阶段飞行器飞行的起始状态。

对于任意阶段，可得到下列动力学方程：

$$\begin{cases} v_{\alpha后}=v_{\alpha前}+a_{\alpha}t \\ v_{r后}=v_{r前}+a_{r}t \\ \beta = \dfrac{\pi}{180}\times 186.6 - \dfrac{\pi}{180}\times 0.1t \\ h_{r后}=h_{r前}-v_{r}t - \dfrac{1}{2}a_{r}t^{2} \\ h_{\alpha后}=h_{\alpha前}-v_{\alpha}t - \dfrac{1}{2}a_{\alpha}t^{2} \end{cases} \tag{20}$$

由牛顿第二定律，可知：

$$a(t) = \frac{T}{m(t)} \tag{21}$$

通过（20）（21）式得到飞行器在该阶段终点时的飞行参数，并以次作为下一阶段的起始飞行参数，以此方式进行迭代。

主减速运动约束条件：

① 近月点速度为$v_{\alpha} = 1.6922km/s$，方向在建立的坐标系中为水平向右。

② 在离地 3km 处，$v_{\alpha} \le 50m/s, v_{r} \le 100m/s$。 。

③ 数值下降位移为：(12 ± 0.2) km。

其中，对于操纵角 β 的方程，可用泰勒方程展开式表示，此处简化只进行三次拟合，表达式为

$$\beta = \sum_{i=0}^{3} a_i t^i \tag{22}$$

为了简化处理，认为推动力只是为了提供制动力，那么将操纵角 β 限定在如下范围内变动：$90^{o} \le \beta \le 270^{o}$，同时假设操纵角是不能快速变化的，该模型认为操纵角在每一阶段的变化幅度在 1° 之内，见图四。

题给推力大小的约束条件为：$1500N \le T \le 7500N$ 。

题目的要求是尽量减少软着陆过程的燃料消耗，可以转化为卫星减少的质量最少，即 $max(m)$ 。而质量减少量的公式为：

$$\Delta m = \frac{Tt}{I_{sq}} \tag{23}$$

利用迭代算法推出每一阶段的初始值，再用 MATLAB 软件求解上述问题（程序代码见附录）。

求得主减速运动曲线长度为 390.66km，在主减速终止位置的水平速度 $v_\alpha = 15.7143m/s$ ， 垂直速度 $v_r = 84.1404m/s$ ，下降高度为 11.889km，运动时间为 430s。

主减速运动的终止位置对月坐标为（$19.51^oW, 44.12^oN$，11.889km）。为求得近月点位置，现已求得近月点与离地约 3km 之间的水平位移以及下降高度，由于月球半径远大于近月点距月球地面的 15km,因此将月心、近月点、着陆点三点处理为扇形的三点，月心仍然是圆心，

半径处理为 $\bar{r} = \frac{15+0}{2} + 1737.013 \approx 1744.513(\text{km})$，

则扇形内夹角：$\frac{390.66}{1744.513} \approx 0.2239(\text{rad}) \approx 12.83^o$ 。

卫星从近月点到着陆点的方向为轨道切线方向，即经度相同，纬度为 $44.12^oN - 12.83^o = 31.29^oN$ 。

得出近月点位置为（$19.51^oW, 31.29^oN, 15km$），卫星在近月点速度为 1.6922km/s，速度方向为近月点在轨道的切线方向。根据近月点，远月点在椭圆轨道相对称的特征，远月点的位置为（$160.49^oE, 31.29^oS, 100km$），卫星在远月点的速度为 1.6139km/s，方向为轨道切线方向。

5.2 问题二

将整体分解为六个状态，依次根据初始值或上一状态的终止值依次求解六个状态的每一状态的最优控制。

5.2.1 着陆准备轨道

此阶段不需要推力。卫星运动曲线为椭圆，月心为其中的一个焦点。由问题一知，卫星在近月点速度为 1.6922km/s，方向为轨道切线方向。远月点的位置为（$160.49^oE, 31.29^oS, 100km$），卫星在远月点的速度为 1.6139km/s，方向为轨道切线方向。

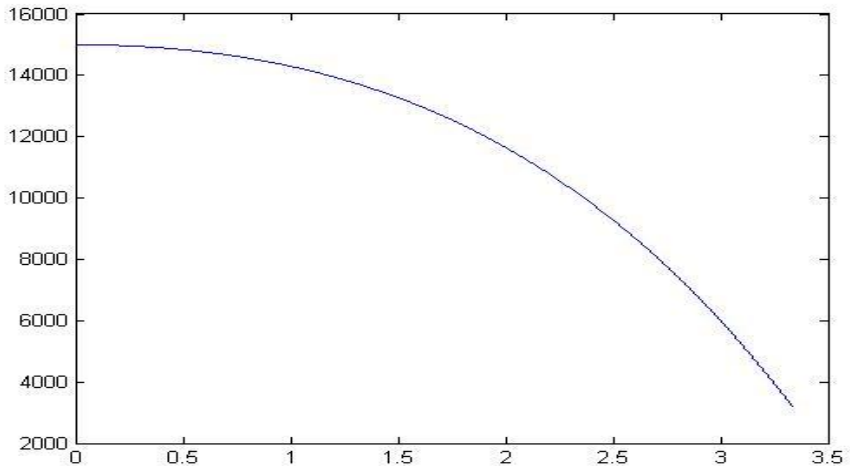
5.2.2 主减速段

由问题一结果知，近月制动点的位置为（$19.51^{o}W,31.30^{O}N,15km$），卫星在近月点速度为 1.6922km/s，速度方向为轨道切线方向。分析方法见问题一的求解，得出：在离地 3km 处时卫星质量为 1303.1kg。

其着陆准备轨道方程：

$$y = 10^{4}(-0.1367x^{2} + 0.1224x + 1.4759)(\mathrm{m}) \tag{24}$$

其中，y 为离地高度 x 的单位为$10^{5}m$ 。



图六　主减速运动拟合曲线图

此阶段控制策略为：推力 T 为 7500N,大小不变，方向实时改变，操纵角 β 随时间的表现为一条具有负斜率的直线，其公式为

$$\beta = \frac{\pi}{180} \times 186.6 - \frac{\pi}{180} \times 0.1t \tag{25}$$。

5.2.3 快速调整段

根据题目要求，从离地 3km 处降到 2.4km 处，水平速度减为 0。对该阶段进行简化，认为推力 T 这一过程中大小不变，方向与水平速度方向相反。由于水平位移小，运动时间少，其比冲（即单位质量的推进剂产生的推力）为 2940m/s，水平速度的速度减少量小，则燃料耗能少，认为卫星质量在这阶段近似不变，即 $m = 1303.1kg$ 。卫星的初始点位置为：（$19.51^{o}W,44.12^{o}N,3.111km$），在 X,Y

坐标系的坐标为（390.66km,3.111km），初始$v_r = 84.1404m/s$，$v_\alpha = 15.7143m/s$， 由物理学知识有：

$$g_{月}=\frac{GM}{r^2} \tag{26}$$

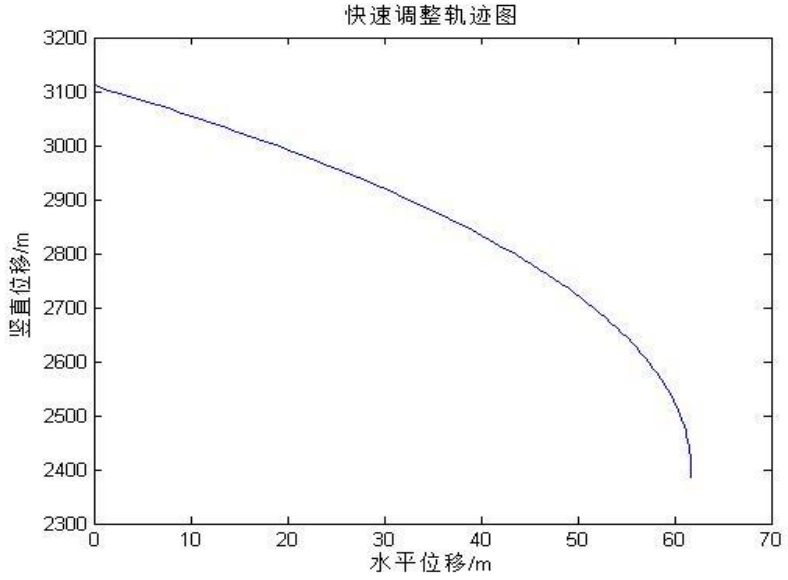
$$竖直方向位移\Delta y = v_r t + \frac{1}{2} g_{月} t^2 \tag{27}$$

$$水平方向位移\Delta x=v_\alpha t - \frac{1}{2} a t^2 \tag{28}$$

$$水平方向速度0 = v_\alpha - at \tag{29}$$

$$水平加速度a = \frac{T}{m} \tag{30}$$

根据上述方程及初始坐标，利用 MATLAB 软件编程（程序代码见附录），并拟合出该状态的运动曲线$y = y(x)$。得出



图七　快速调整阶段的拟合曲线图

快速调整阶段运动时间为 7.86s，竖直方向速度$v_r = 93.3m/s$，水平位移较月球半径很小，终止点在月面上的垂直投影点不变，即终止点位置为（$19.51^oW,44.12^oN,2.4km$）。推力 T=2602.1N，与水平速度方向时刻相反。卫星质量的减少量$\Delta m = \frac{Tt}{I_{sq}} = 6.95kg$，则终止位置时卫星质量为$m = 1296.15kg$。

快速准备轨道曲线方程：

$$y = 10^3(-0.0002x^2 + 0.001x + 3.0641)\,(\mathrm{m}) \tag{31}$$

X 的单位为 m。

最优控制策略：推力 T=2602.1N，推力方向与水平速度方向始终相反。

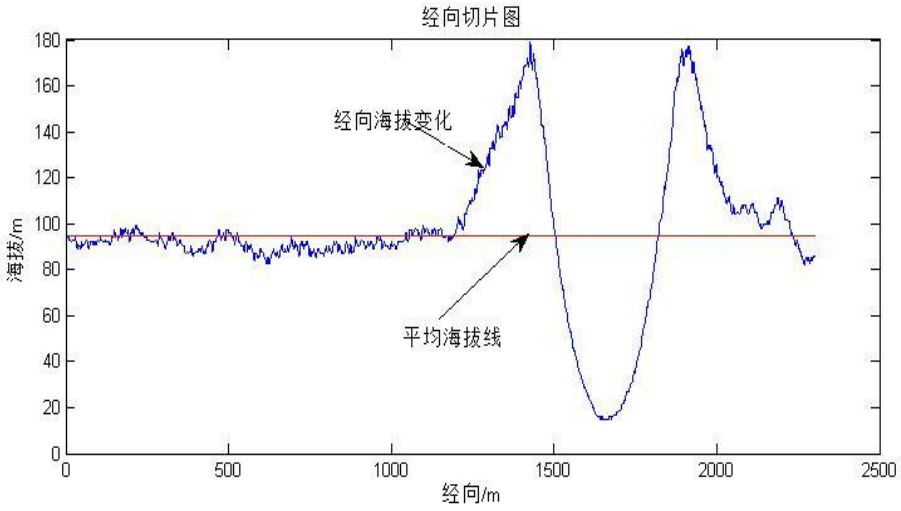
5.2.4 粗避障段

5.2.4.1 安全着陆点选择的螺旋模型

安全着陆区的选取[4]，采用从预着陆点开始螺旋前进搜索的方法(图九)，图八中每个单元格代表 3×3 的单元块。单元块中心每移动一步，所选取的单元块如图九所示。根据坡度的阈值的限制条件，进行螺旋搜索，找到符合安全着陆要求的着陆区域，确定安全着陆点。

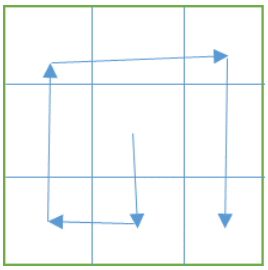
卫星在 2.4km 要进行自主粗步避开大陨石坑，即需要避免落在较大面积的陨石坑及坡度较陡的地方。卫星扫描月球表面，通过用 MATLAB 提取附件三中距月面 2400m 处的数字高程图的灰度值。考虑到卫星的占地面积大小，选取灰度值矩阵中 3×3 单元块（九个数据）为卫星的占地面积。

利用图片中所有灰度值的平均值，用平均值表示平均海拔高度。然后求出 3×3 的单元块灰度值的平均值和方差，如图十，用该平均值表示这一区域所在海拔高度，用该方差表示这一区域的坡度。在判断预着陆区域是否为安全着陆点时，首先考虑该区域的海拔高度，如果它在平均海拔高度一定范围内波动，那么认为它是安全的。但是考虑到如下图八情况：

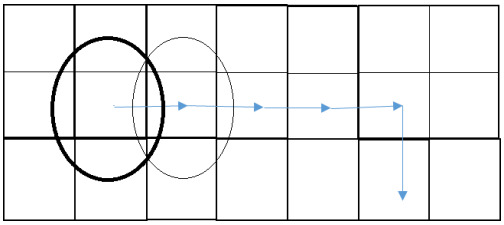


图八 粗避障的经向切片图

对于坑的坡面上，可能存在某些点，由于其左右海拔高度差相互抵消，导致该区域的海拔在安全的海拔波动范围内，需要利用方差剔除这样的点。如果该区域的方差在一定范围内，那么认为该区域是安全着陆点。平均值和方差的波动范围阈值的确定在设定坡度阈值的模型中解决。



图九：螺旋搜索方法示意图



图十：3×3 单元块的随搜索的选取

5.2.4.2 设定坡度阈值的模型

设定可着陆区域的坡度的阈值的算法[3] 具体步骤为：

① 对每一幅降落图像，采用离散二维熵双阈值分割方法对图像进行分割。离散二维熵 H 定义如下式：

$$H = -\sum_{i}\sum_{j} p_{ij} \lg p_{ij} \tag{32}$$

式中 p_{ij} 为点灰度——区域均值对(i, j)发生的概率:

$$p_{ij} = \frac{n_{ij}}{N \times M} \tag{33}$$

NxM 为图像大小，n_{ij} 为图像中心点灰度 i 区域均值为 j 的像素点的个数。

以原始灰度图像((L 个灰度级)中每个像素及其 8 邻域的 8 个像素为 1 个区域，计算出区域灰度均值图像，这样原始图像中的每 1 个像素点都对应于 1 个点灰度——区域灰度均值对，进行处理得到降落图像的二维直方图。

② 用最优阈值选择方法选择阈值。

设阈值为(s ， t)，定义熵的判别函数如下式所示:

$$\varphi(\mathrm{s,t}) = \lg[\mathrm{P}_A(1-\mathrm{P}_A)] + \frac{H_A}{P_A} + \frac{H_L - H_A}{1-P_A} \tag{34}$$

式中, P_A 为 A 区总概率； H_A 为 A 区二维熵； H_L 为整个图像的二维熵，即：

$$P_A = \sum_{i=1}^{s}\sum_{j=1}^{t} p_{ij} \tag{35}$$

$$H_A = -\sum_{i=1}^{s}\sum_{j=1}^{t} p_{ij}\lg p_{ij} \tag{36}$$

$$H_L = -\sum_{i=1}^{L}\sum_{j=1}^{L} p_{ij}\lg p_{ij} \tag{37}$$

计算使得式(34)取最大值的灰度值即为最佳阈值。

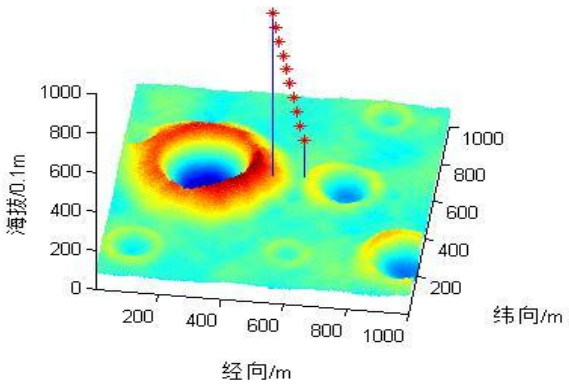
5.2.4.3 粗避障模型的求解

通过上述方案，得到粗避障过程不需要水平移动，是垂直下落过程。为简化模型，假设该过程中卫星质量不变，做匀加速直线运动到达着陆点 100 米上方处。

利用 MATLAB 软件编程（程序代码放在附录中），得出推力 $T = 4370.9N$，$t = 50.94s$ ，再根据推力求得卫星质量减少量 $\Delta m = 76.77kg$。运动方向为竖直方向。

5.2.5 精避障段

较于粗避障模型，分析过程一样，区别在于坡度的阈值大小的设定，仍根据粗避障模型求解。而且精避障的运动约束条件：下降到 30 米处的水平速度为 0，从而判断出最近的可着陆点，如图十。



图十：精避障段的运动曲线三维示意图

利用 MATLAB 软件编程（程序代码放在附录中），$t = 9.35s$，$v_r = 14.96m/s$，$v_\alpha = 0$,T=0。卫星质量仍为 1219.38kg。

5.2.6 缓冲下降阶段

距地面 30 米处已于着陆点上方，高度为 4 米内的运动为自由落体下降，该过程目的是减缓竖直向下速度，使得到达距地面速度较小，从而实现软着陆。

运用快速调整阶段的运动方程，仍然假设运动中卫星质量不变，计算得出 T=7196N，t=3.48s,再推出 $\Delta m = 8.62kg$，即到着陆点时，卫星质量 $m = 1210.76kg$。运动方向为竖直向下。

5.3 问题三

5.3.1 误差分析

在文献【5】中，卫星软着陆过程用时约 685 s，各模式的维持时间分别为:主减速模式 487 s,快速调整模式 16s，接近模式 125 s,悬停模式 16 s,缓速下降模式 19 s,避障模式 22s。

本文中，主减速模式 430s，快速调整模式 7.86s,粗避障模式 50.94s，精避障模式 9.35s。

误差分析：任何一个航天计划的实施都依赖于对飞行过程中误差源的详细分析。本文建立的模型误差来源，一方面是动力学方程的简化，另一方面动力方程组在迭

代过程中误差的累积。动力学模型的误差主要是该模型忽略了天体引力。

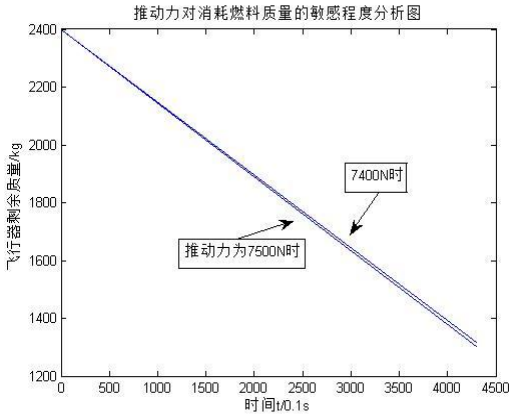
在计算近近月点速度 v 的时候，我们忽略了不同经纬度月球半径不同的影响，对月球的扁率 e 也并未加入模型计算之中。

在着陆轨道的主减速段，我们选择了定常推力的模型，虽然简化了问题，但也出现了相关误差。在进入主减速段之前，我们认为飞行器仰角 β=90°，实际仰角初始值为 85°，这也造成了模型的求解误差。在求解过程中对 β 角进行三次多项式拟合，求解过程中积累误差。

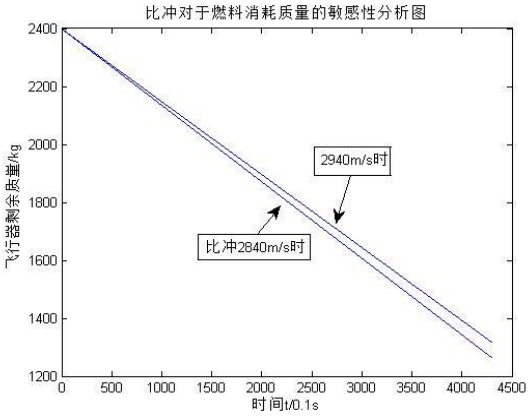
在求解定常推力模型的过程中，利用无限分割，直线逼近曲线的方法，进行动力方程迭代，给予的初始值中的偏差部分被放大，影响结果的准确性。

5.3.2 敏感性分析

敏感性分析：选取影响燃料消耗的变量发动机比冲 I，推动力大小 T 和方向 β，对其做敏感性分析，即改变变量，观察燃料消耗量的变化。定义：用相对改变量衡量参数对结果的敏感程度。



图十一：推动力的敏感性分析图



图十二：比冲的敏感性分析图

对于推力大小 T 对燃料消耗质量 m 的敏感度记作 $S_m(T,m)$ 。根据定义：

$$S_m(T,m) = \frac{\Delta m / m}{\Delta T / T} \tag{38}$$

当推动力 T 变化量为 1.33%时，燃料消耗质量变化为 1.96%，敏感度为 1.471%。

当比冲 I_{sq} 变化量为 3.4%时，燃料消耗质量变化为 33.31%，敏感度为 0.154%。

当操纵角 β 变化量为 0.54%时，燃料消耗质量变化为 1.96%，敏感度 0.387%。

6 模型的评价

（1）该模型利用了差分动力方程组求解主减速阶段的定常推力软着陆问题，求解简单，能够达到燃料消耗最少的目标，有较高的推广价值。

（2）利用迭代方法，对系统的初值及误差较为敏感。

（3）在着陆避障中运用了螺旋搜索，能够选取合适登陆地区。

参考文献

【1】于永刚，行星在椭圆轨道上运动的速度和能量，http://wenku.baidu.com/link?url=voPg1GHpOB5yC2Wgd_i3FVddjV4RtYJtVmtW4864xS_HlHRQ4V7g5T6H28f-I7-qW5g2X9V7gXRO38LUCFz-98funKljKmT-7Yz7kfoovDK&qq-pf-to=pcqq.discussion ，2014 年 9 月 12 日。

【2】王暾，李俊峰，崔乃刚等，登月飞行器软着陆轨道的遗传算法优化，清华大学学报(自然科学版)，第 43 卷：1056-1059 页 ，2003 年。

【3】王海涛，马建华，基于降落图像的安全着陆点选择技术研究，航天返回与遥感，第 33 卷第 4 期：13-18 页，2012 年。

【4】张洪华，梁俊，黄翔宇等，嫦娥三号自主避障软着陆控制技术，中国科学，第 44 卷第 6 期:559-568 页，2014 年。

【5】张洪华，关轶峰，黄翔宇等，嫦娥三号着陆器动力下降的制导导航与控制，中国科学，第 44 卷第 4 期:377 -384 页，2014 年。

附录

1.1 求径向高度变化

```
clc;
close all;
clear all;
G=6.67*10^(-11);
M=7.3477*10^(22);
t=0.1;
b=pi/180*186.6;
ve=2940;
v00=[1692.2];
vrr=[0];
hrr=[15000];
h00=[0];
T=0;
x=0;
y=0;
FF=0;
```

```
yy=0;
for F=1500:100:7500
    m=2400;
    tt=0;
    r=1752000;
    hr=0;
    h0=0;
    a=F/m;
    mm=0;
    vr=0;
    v0=1692.2;
    while(tt<430)
        ar=-G*M/(r^2)+v0^2/r+a*sin(b);
        a0=a*cos(b);
        r=r+vr*t+0.5*ar*t^2;
        m=m-F*t/ve;
        a=F/m;
        v0=v0+a0*t;
        vr=vr+ar*t;
        b=pi/180*186.6-pi/180*0.1*tt;
        hr=hr-vr*t-0.5*ar*t^2;
        h0=h0+v0*t+0.5*a0*t^2;
        tt=tt+t;
    end
    if m>mm
        mm=m;
        T=tt;
        FF=F;
        x=h0;
        y=v0;
        yy=mm;
    end
end
F=7500;
   m=2400;
    tt=0;
    r=1752000;
    hr=15000;
    h0=0;
    a=F/m;
    mm=0;
    vr=0;
    v0=1692.2;
while(v0>100 & hrr>3200 )
        ar=-G*M/(r^2)+v0^2/r+a*sin(b);
        a0=a*cos(b);
        r=r+vr*t+0.5*ar*t^2;
        m=m-F*t/ve;
        a=F/m;
        v0=v0+a0*t;
        v00=[v00,v0];
        vrr=[vrr,vr];
        hrr=[hrr,hr];
        h00=[h00,h0];
```

```
        vr=vr+ar*t;
        b=pi/180*186-pi/180*0.1*tt;
        hr=hr+vr*t+0.5*ar*t^2;
        h0=h0+v0*t+0.5*a0*t^2;
        tt=tt+t;
end
plot(hrr)
```

1.2 求切向位移时间曲线

```
clc;
close all;
clear all;
G=6.67*10^(-11);
M=7.3477*10^(22);
t=0.1;
b=pi/180*186.6;
ve=2940;
v00=[1692.2];
vrr=[0];
hrr=[15000];
h00=[0];
T=0;
x=0;
y=0;
FF=0;
yy=0;
for F=1500:100:7500
    m=2400;
    tt=0;
    r=1752000;
    hr=0;
    h0=0;
    a=F/m;
    mm=0;
    vr=0;
    v0=1692.2;
    while(v0>100 & hrr>3200)
        ar=-G*M/(r^2)+v0^2/r+a*sin(b);
        a0=a*cos(b);
        r=r+vr*t+0.5*ar*t^2;
        m=m-F*t/ve;
        a=F/m;
        v0=v0+a0*t;
        vr=vr+ar*t;
        b=pi/180*186.6-pi/180*0.1*tt;
        hr=hr-vr*t-0.5*ar*t^2;
        h0=h0+v0*t+0.5*a0*t^2;
        tt=tt+t;
    end
    if m>mm
        mm=m;
        T=tt;
        FF=F;
```

```
            x=h0;
            y=v0;
            yy=mm;
        end
end
F=7500;
   m=2400;
        tt=0;
        r=1752000;
        hr=15000;
        h0=0;
        a=F/m;
        mm=0;
        vr=0;
        v0=1692.2;
while(v0>100 & hrr>3200 )
            ar=-G*M/(r^2)+v0^2/r+a*sin(b);
            a0=a*cos(b);
            r=r+vr*t+0.5*ar*t^2;
            m=m-F*t/ve;
            a=F/m;
            v0=v0+a0*t;
            v00=[v00,v0];
            vrr=[vrr,vr];
            hrr=[hrr,hr];
            h00=[h00,h0];
            vr=vr+ar*t;
            b=pi/180*186-pi/180*0.1*tt;
            hr=hr+vr*t+0.5*ar*t^2;
            h0=h0+v0*t+0.5*a0*t^2;
            tt=tt+t;
end
plot(h00)
```

1.3 求径向速度时间曲线

```
clc;
close all;
clear all;
G=6.67*10^(-11);
M=7.3477*10^(22);
t=0.1;
b=pi/180*186.6;
ve=2940;
v00=[1692.2];
vrr=[0];
hrr=[15000];
h00=[0];
T=0;
x=0;
y=0;
FF=0;
yy=0;
for F=1500:100:7500
```

```
        m=2400;
        tt=0;
        r=1752000;
        hr=0;
        h0=0;
        a=F/m;
        mm=0;
        vr=0;
        v0=1692.2;
        while(tt<430)
            ar=-G*M/(r^2)+v0^2/r+a*sin(b);
            a0=a*cos(b);
            r=r+vr*t+0.5*ar*t^2;
            m=m-F*t/ve;
            a=F/m;
            v0=v0+a0*t;
            vr=vr+ar*t;
            b=pi/180*186.6-pi/180*0.1*tt;
            hr=hr-vr*t-0.5*ar*t^2;
            h0=h0+v0*t+0.5*a0*t^2;
            tt=tt+t;
        end
        if m>mm
            mm=m;
            T=tt;
            FF=F;
            x=h0;
            y=v0;
            yy=mm;
        end
end
F=7500;
   m=2400;
        tt=0;
        r=1752000;
        hr=15000;
        h0=0;
        a=F/m;
        mm=0;
        vr=0;
        v0=1692.2;
while(v0>100 & hrr>3200 )
            ar=-G*M/(r^2)+v0^2/r+a*sin(b);
            a0=a*cos(b);
            r=r+vr*t+0.5*ar*t^2;
            m=m-F*t/ve;
            a=F/m;
            v0=v0+a0*t;
            v00=[v00,v0];
            vrr=[vrr,vr];
            hrr=[hrr,hr];
            h00=[h00,h0];
            vr=vr+ar*t;
            b=pi/180*186-pi/180*0.1*tt;
```

```
        hr=hr+vr*t+0.5*ar*t^2;
        h0=h0+v0*t+0.5*a0*t^2;
        tt=tt+t;
end
plot(vrr)
```

1.4 求切向速度时间曲线

```
clc;
close all;
clear all;
G=6.67*10^(-11);
M=7.3477*10^(22);
t=0.1;
b=pi/180*186.6;
ve=2940;
v00=[1692.2];
vrr=[0];
hrr=[15000];
h00=[0];
T=0;
x=0;
y=0;
FF=0;
yy=0;
for F=1500:100:7500
    m=2400;
    tt=0;
    r=1752000;
    hr=0;
    h0=0;
    a=F/m;
    mm=0;
    vr=0;
    v0=1692.2;
    while(tt<430)
        ar=-G*M/(r^2)+v0^2/r+a*sin(b);
        a0=a*cos(b);
        r=r+vr*t+0.5*ar*t^2;
        m=m-F*t/ve;
        a=F/m;
        v0=v0+a0*t;
        vr=vr+ar*t;
        b=pi/180*186.6-pi/180*0.1*tt;
        hr=hr-vr*t-0.5*ar*t^2;
        h0=h0+v0*t+0.5*a0*t^2;
        tt=tt+t;
    end
    if m>mm
        mm=m;
        T=tt;
        FF=F;
        x=h0;
        y=v0;
```

```
            yy=mm;
        end
end
F=7500;
   m=2400;
        tt=0;
        r=1752000;
        hr=15000;
        h0=0;
        a=F/m;
        mm=0;
        vr=0;
        v0=1692.2;
while(v0>100 & hrr>3200 )
            ar=-G*M/(r^2)+v0^2/r+a*sin(b);
            a0=a*cos(b);
            r=r+vr*t+0.5*ar*t^2;
            m=m-F*t/ve;
            a=F/m;
            v0=v0+a0*t;
            v00=[v00,v0];
            vrr=[vrr,vr];
            hrr=[hrr,hr];
            h00=[h00,h0];
            vr=vr+ar*t;
            b=pi/180*186-pi/180*0.1*tt;
            hr=hr+vr*t+0.5*ar*t^2;
            h0=h0+v0*t+0.5*a0*t^2;
            tt=tt+t;
end
plot(v00)
```

2.1 避障算法　　求方差平均值

```
clc;
clear all;
close all;
a=imread('D:\cu.tif');
aver=0;
s=0;
b=zeros(2300,2300);
bb=zeros(5,5);
av=zeros(2300,2300);
ss=zeros(2300,2300);
for i=1:2300
        for j=1:2300
            b(i,j)=a(i,j);
        end
end
aver=mean(mean(b));
s=var(b(:));
for i=3:2298
        for j=3:2298
            for ii=-2:2
                for jj=-2:2
```

```
                bb(3+ii,3+jj)=b(i+ii,j+jj);
            end
        end
        av(i,j)=mean(mean(bb));
        ss(i,j)=var(bb(:));
    end
end
```

2.2 避障算法　求最近安全着陆点

```
clc;
x=2300/2;
y=2300/2;
bounds1=10;
bounds2=0.5;
locate=0;
z=3;
dx=zeros(1,8);
    if av(x,y)<aver+bounds1 & av(x,y)>aver-bounds1 & ss(x,y)<bounds2
        locate=1;
        break;
    else
        if av(x,y+1)<aver+bounds1 & av(x,y+1)>aver-bounds1 & ss(x,y+1)<bounds2
            locate=1;
            break;
            y=y+1;
        else
            if av(x+1,y+1)<aver+bounds1 & av(x+1,y+1)>aver-bounds1 & ss(x+1,y+1)<bounds2
                locate=1;
                break;
                x=x+1;
                y=y+1;
            else
                if av(x+1,y)<aver+bounds1 & av(x+1,y)>aver-bounds1 & ss(x+1,y)<bounds2
                    locate=1;
                    break;
                    x=x+1;
                else
                    if av(x+1,y-1)<aver+bounds1 & av(x+1,y-1)>aver-bounds1 &
ss(x+1,y-1)<bounds2
                        locate=1;
                        break;
                        y=y-1;
                        x=x-1;
                    else
                        if av(x,y-1)<aver+bounds1 & av(x,y-1)>aver-bounds1 &
ss(x,y-1)<bounds2
                            locate=1;
                            break;
                            y=y-1;
                        else
                            if av(x-1,y-1)<aver+bounds1 & av(x-1,y-1)>aver-bounds1 &
ss(x-1,y-1)<bounds2
                                locate=1;
```

```
                                        break;
                                        x=x-1;
                                        y=y-1;
                                    else
                                        x=x-1;
                                        y=y-1;
                                    end
                                end
                            end
                        end
                    end
                end
            end
x
y
while(locate==0)
    for i=1:4
        for j=1:z
            if i==1 & locate==0
                y=y+1;
                if av(x,y)<aver+bounds1 & av(x,y)>aver-bounds1 & ss(x,y)<bounds2
                    locate=1;
                    break;
                end
            end
            if i==2 & locate==0
                x=x+1;
                if av(x,y)<aver+bounds1 & av(x,y)>aver-bounds1 & ss(x,y)<bounds2
                    locate=1;
                    break;
                end
            end
            if i==3 & locate==0
                y=y-1;
                if av(x,y)<aver+bounds1 & av(x,y)>aver-bounds1 & ss(x,y)<bounds2
                    locate=1;
                    break;
                end
             end
             if i==4 & locate==0
                x=x-1;
                if av(x,y)<aver+bounds1 & av(x,y)>aver-bounds1 & ss(x,y)<bounds2
                    locate=1;
                    break;
                end
             end
        end
    end
    z=z+1;
end
x
y
```

2.3　避障算法　起点终点

```
tu=zeros(1000,1000);
for i=1:1000
    for j=1:1000
        tu(i,j)=b(i,j);
    end
end
for i=-10:10
    for j=-10:10
        tu(1000/2+i,1000/2+j)=0;
        tu(592+i,561+j)=0;
    end
end
imshow(uint8(tu)
```