# 3.1 直线的扫描转换--中点画线算法

## 直线扫描转换的速度要求

尽量使用整数加减法，避免乘、除、开方、三角等复杂运算，避免取整运算。

y=kx+b(浮点乘、浮点加、取整)

## 中点画线算法的前提

起点：P0(x0,y0)，终点：P1(x1,y1) 都是整数

0≤k≤1，则|△x|≥|△y|，x0<x1,y0<y1

## 增量算法

xi+1=xi+1(整数)

yi+1=yi+k（浮点加，取整）

## 中点误差项



yi+1= yi +1 ----中点在直线下，di<0

yi+1= yi ----中点在直线上, di≥0

di+1=di+1-k , di<0

di+1=di-k , di≥0



di+1中仍有浮点加

令

fi=2di△x

## 整数中点误差项



xi+1=xi+1

yi+1= yi +1 ----中点在直线下，fi<0

yi+1= yi ----中点在直线上, fi≥0

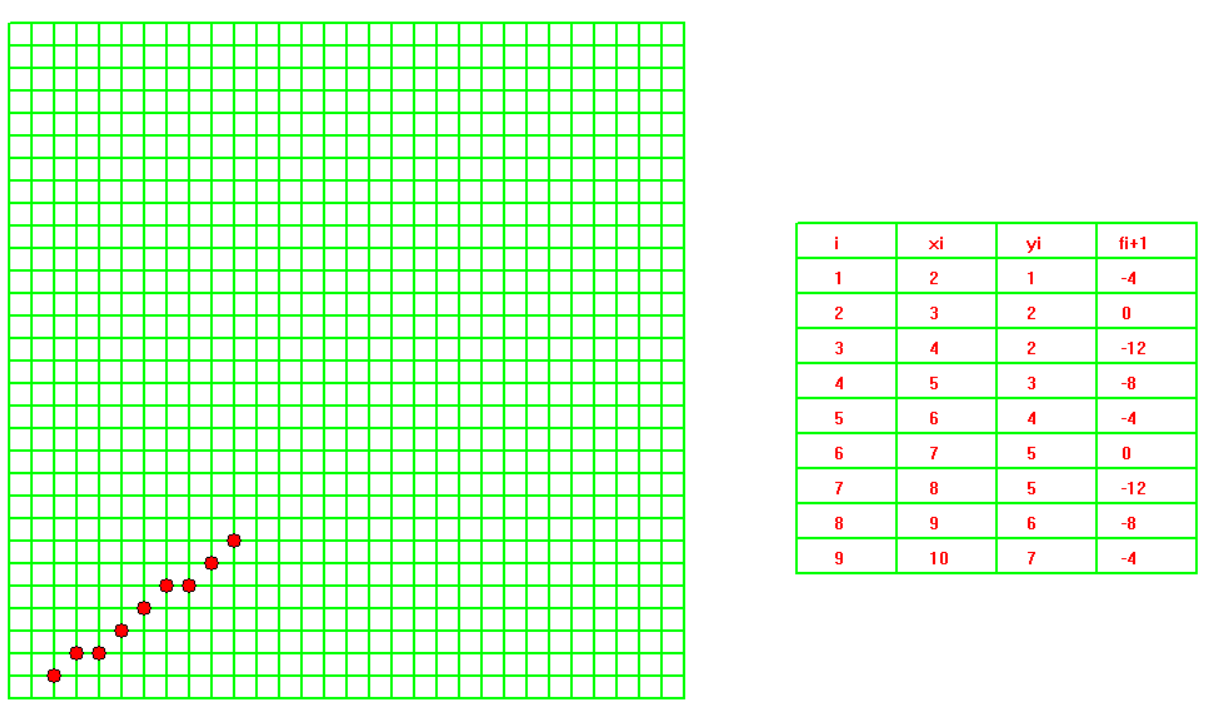
## 算法

算法与公式对照。

* x=x0;y=y0;
* dx=x1-x0;dy=y1-y0;
* f=dx-2\*dy;//式(3-7)
* for (i=1; i<=dx+1; i++)
* {setpixel(x,y,color); //画点，统一各变量值的位置
* x=x+1;
* if (f<0)
* {y=y+1; //式（3-3）
* f=f+2\*dx;｝//式（3-8）
* f=f-2\*dy;
* }

## 实例

起点（2，1），终点（10，7），按算法给出每一点各项变量值。



|  |  |  |  |
| --- | --- | --- | --- |
| i | xi | yi | Fi+1 |
| 1 | 2 | 1 | -4 |
| 2 | 3 | 2 | 0 |
| 3 | 4 | 2 | -12 |
| 4 | 5 | 3 | -8 |
| 5 | 6 | 4 | -4 |
| 6 | 7 | 5 | 0 |
| 7 | 8 | 5 | -12 |
| 8 | 9 | 6 | -8 |
| 9 | 10 | 7 | -4 |

10

## 习题

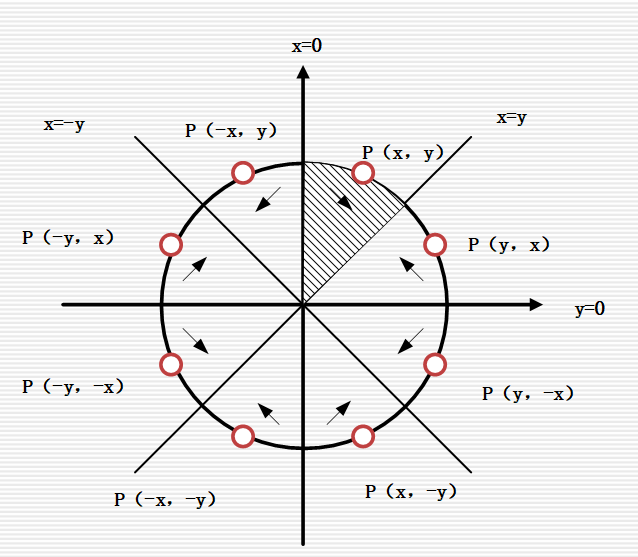
起点（3，2），终点（9，7），给出每点的i、x、y、f值。

# 3.2圆的扫描转换

## 根据方程画圆

假设圆心在原点，半径为Ｒ：

## 圆的对称性



只计算第一象限上半区间的点，通过对称性画出其余的点。

先假设圆心在原点，实现算法后，再推广到所有情况。

## 增量算法

xi+1=xi+1(整数)

yi+1={ yi , yi -1} (yi是递减的，递减幅度不超过1)

## 中点误差项



yi+1= yi 　----中点在直线下，di<0

yi+1= yi-1 ----中点在直线上, di≥0

*di*+1= *di*+2 *xi*+3*di*<0

*di*+1= *di*+2 (*xi*- *yi*)+5*di*≥0

*d*0= 1.25-R1-R

## 算法

void CTestView::MBCircle(double R,CDC \*pDC)//圆中点中点画线算法

{

double x,y,d;

d=1.25-R;x=0;y=R; //可用d=1-R计算d

for(x=0;x<=y;x++)

{

CirclePoint(x,y,pDC);//画圆子函数，统一各变量值的位置

if (d<0)

d+=2\*x+3;

else

{

d+=2\*(x-y)+5;

y--;

}

}

}

## 实例

给出半径R=10的1/8圆的生成过程，即每点的x、y、d值，画出这些点。