Pinwheel Puzzle

Ever try to hit something with a spinning stick? Ever try to hit something with a spinning stick while spinning on the end of a spinning stick? What about hitting something with a spinning stick while on a spinning stick on a spinning stick on a spinning stick on a spinning stick?

Pinwheel puzzle gets pretty crazy pretty quickly. The key is remembering that the speed your last line is spinning remains constant the moment you touch your goal. If you're clever you can have the motion of two or more lines eventually resolve themselves into less motion, but once you hit a goal too fast there's no turning back.

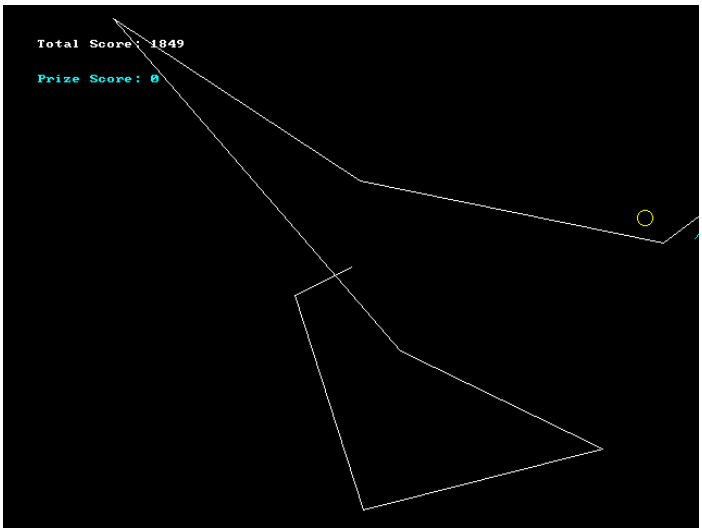
Pinwheel Puzzle is written by Slartibartfast on the Allegro.cc forums.

// pinwheelpuzzle.cpp listing begins:

```
#include<math.h>
#include<allegro.h>
volatile long speed_counter = 0;
void increment_speed_counter() {
  speed_counter++;
}
END_OF_FUNCTION(increment_speed_counter);

BITMAP *BUFFER;

class Circle {
  private:
    double x;
    double y;
    double angle;
    double len;
    double vel;
    double tx;
    double ty;
    long score;
    Circle *next;
  public:
    Circle(double X,double Y,double ANGLE,double LEN,double VEL,
      double TX,double TY) {
      x=X; y=Y; angle=ANGLE; len=LEN; vel=VEL; next=NULL;
      tx=TX; ty=TY; score=1000;
    }
    Circle() {
      x=320; y=240; angle=0; len=20; vel=0.01; next=NULL;
      tx=320; ty=300; score=1000;
    }
    void HitLeft() {
      if (next==NULL)
        vel+=0.001;
      else
        next->HitLeft();
    }
    void HitRight() {
      if (next==NULL)
        vel-=0.001;
      else
        next->HitRight();
    }
    void HitUp() {
      if (next==NULL)
        len+=1.5;
      else
        next->HitUp();
    }
    void HitDown() {
      if (next==NULL)
        len-=1.5;
      else
        next->HitDown();
    }
    void Logic(int xn,int yn) {
      x=xn;
      y=yn;
      angle+=vel;
      if (next==NULL) {
        line(BUFFER,x,y,x+sin(angle)*len,y+cos(angle)*len,makecol(0,255,255));
```



// Listing continued on next page…

// Listing continued from previous page

```
        circle(BUFFER,tx,ty,7,makecol(255,255,0));
        if (sqrt( (x+sin(angle)*len-tx)*(x+sin(angle)*len-tx)
            + (y+cos(angle)*len-ty)*(y+cos(angle)*len-ty) ) < 7) {
          next=new Circle(x+sin(angle)*len,y+cos(angle)
                    *len,angle,len,vel,rand()%640,rand()%480);
        }
        if (score>0)
          score-=1;
      } else {
        line(BUFFER,x,y,x+sin(angle)*len,y+cos(angle)*len,makecol(255,255,255));
        next->Logic(x+sin(angle)*len,y+cos(angle)*len);
      }
    }
    void Score(long S) {
      if (next==NULL) {
        textprintf_ex(BUFFER, font, 32, 32, makecol(255, 255, 255),-1,
          "Total Score: %ld", S);
        textprintf_ex(BUFFER, font, 32, 64, makecol(0, 255, 255),-1,
          "Prize Score: %ld", score);
      } else {
        next->Score(S+score);
      }
    }
};

int main(int argc, char *argv[]) {
  allegro_init();
  install_mouse();
  install_keyboard();
  install_timer();
  LOCK_VARIABLE(speed_counter);
  LOCK_FUNCTION(increment_speed_counter)
  install_int_ex(increment_speed_counter, BPS_TO_TIMER(60));
  set_color_depth(desktop_color_depth());
  set_gfx_mode(GFX_AUTODETECT, 640,480,0,0);
  BUFFER=create_bitmap(640,480);
  srand(time(NULL));

  Circle L;

  speed_counter=1;
  while (!key[KEY_ESC]) {
    while ((speed_counter>0)&&(!key[KEY_ESC])) {
      if (key[KEY_UP])
      L.HitUp();
      if (key[KEY_DOWN])
      L.HitDown();
      if (key[KEY_LEFT])
      L.HitLeft();
      if (key[KEY_RIGHT])
      L.HitRight();
      L.Score(0);
      L.Logic(320,240);
      draw_sprite(screen,BUFFER,0,0);
      clear_bitmap(BUFFER);
      speed_counter--;
    }
  }
  return 0;
}
END_OF_MAIN()
```