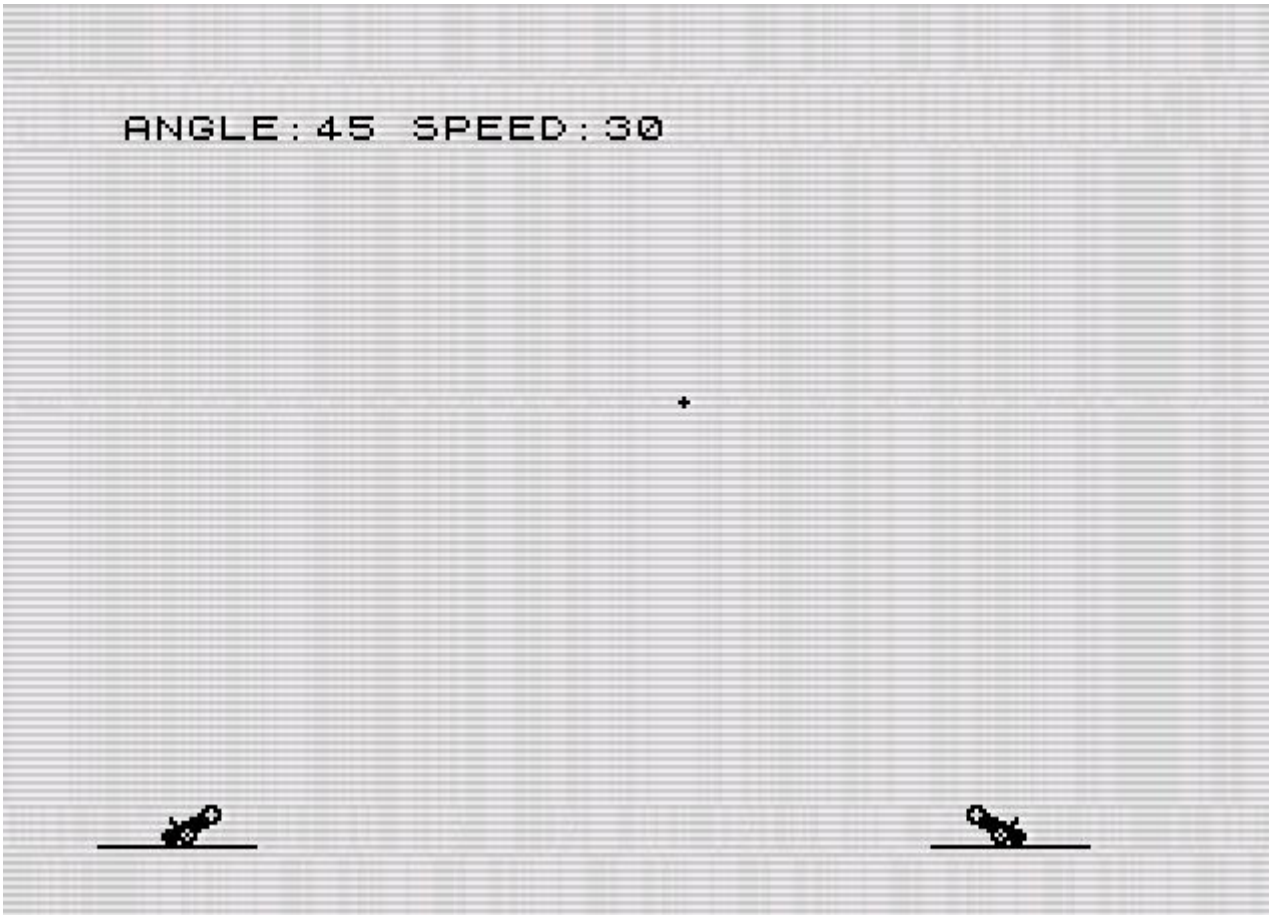
Artillery Duel



A new method of displaying screens. A combined Blocky/Pong/Marble Racer screen.
A classic game where you enter an angle and speed to shoot a cannonball.
The drawing is better than in Claypigeonshooter, but fully correct Bresenhammovement was impossible in 1K. In fact the entire code was really tight. To get a bit of surprise windeffect is added after releasing enough bytes in the collisiondetection.
I managed to reduce the stack so it is no longer visible on the screen.

```
; Artillery Duel
; Game 29 in 1K hires for the ZX81

; dy=speed*sin(angle)
; dx=speed*cos(angle)
; always wind effect
; stack is shortened by saving angle in memory
; stack 2 bytes shorter now, program same size

? * TORNADO *

            ORG  #4009              ;#4009
            DUMP 49161

            JP   init               ; start of program
d_file      DEFW dfile
dfcc        DEFW dfile+1
var         DEFW vars
dest        DEFW 0
```

```
eline       DEFW last
chadd       DEFW last-1
xptr        DEFW 0
stkbot      DEFW last
            DEFW last               ; memory above reused for data

berg        DEFB 0
mem         DEFW 0

code        DEFB 0                  ; altered by init
            DEFB 2                  ; altered by init
            JR   nxtlin

lastk       DEFB 255,255,255        ; used by ZX81
margin      DEFB 55                 ; used by ZX81

nxtlin      DEFW basic              ; altered by init
            DEFB 0                  ; altered by init
            RET

flagx       DEFB 0                  ; x
strlen      DEFW 0

taddr       DEFW 3213

seed        DEFW 0
frames      DEFW 65535              ; used by ZX81
coords      DEFB 0,0
prcc        DEFB 188
sposn       DEFB 33,24
cdflag      DEFB 64                 ; fixed value

; A = angle, C = speed, HL = SIN or COS table
fspeed      LD   A,0
            INC  A                  ; 1 more for CARRY
fsp         INC  HL
            SUB  5
            JR   NC,fsp
            LD   A,(HL)             ; fetch SIN or COS
            LD   HL,0
            LD   D,H
            LD   E,C                ; set speed in DE
mulspeed    ADD  HL,DE
            DEC  A
            JR   NZ,mulspeed        ; speed * SIN or COS
            ADD  HL,HL
            JR   code               ; saves 3 bytes, set by init

readkey     LD   E,2
            XOR  A
            LD   (hit+1),A          ; use A for reset
keyblp      LD   D,A                ; previous value
            ADD  A,A                ; *2
            ADD  A,A                ; *4
            ADD  A,D                ; *5
            ADD  A,A                ; *10
            LD   D,A                ; save 10*previous
reread      LD   (HL),128           ; inverse space
            LD   BC,(lastk)         ; fetch key
            LD   A,C
ktest       CP   0
            JR   Z,reread           ; key kept down
            LD   (ktest+1),A        ; also debounce
            INC  A
```

```
            JR   Z,reread          ; no key pressed

            PUSH HL
            PUSH DE
            CALL #7BD              ; translate to ascii
            LD   A,(HL)
            POP  DE
            POP  HL
            LD   (HL),A            ; show each key also non-nr
            SUB  28
            CP   10
            JR   NC,reread         ; only numbers, clear field
            ADD  A,D               ; add tens to units
            INC  HL                ; next position
            DEC  E
            JR   NZ,keyblp         ; 2 numbers
            RET                    ; A hold 2 digit number

start       LD   A,%10111111       ; start of new game
            IN   A,(254)
            RRA                    ; Newline to Carry
            JR   C,start           ; no NL pressed

            CALL rnd
            LD   (cl+1),A          ; random height left cannon
            CALL rnd
            LD   (cr+1),A          ; random height right cannon
            CALL rnd
            AND  3                 ; 0 / 1 / 2 / 3
            SUB  2                 ; -2 / -1 / 0 / 1
setwind     LD   (wind+1),A        ; set windeffect

gameloop    LD   A,245
xpos        SUB  218
            LD   (xpos+1),A
            LD   B,A               ; set x-pos

            LD   A,(endx+1)
            CPL
            LD   (endx+1),A        ; move left or right with ball
            LD   A,(cl+1)          ; l=ypos left
            JP   P,lstart
            LD   A,(cr+1)          ; y-pos right
lstart      DEC  A                 ; y-cannonball = y-cannon - 1
            LD   C,A
            PUSH BC                ; B only for save
            CALL makescr           ; show loaded cannon

            LD   HL,angle
            CALL readkey           ; read angle
            LD   (fspeed+1),A      ; save angle

            LD   L,speed*256/256
            CALL readkey           ; read speed
            LD   C,A               ; set speed

            LD   HL,sin-1          ; calculate dy with sinus
            CALL fspeed
            LD   (dy+1),A          ; set vertical speed
            LD   HL,cos-1          ; calculate dx with cosinus
            CALL fspeed
            LD   (dx+1),A          ; set horizontal speed
            POP  BC                ; original Y and X cannon
```

```
dylp        LD   HL,dy+1
            LD   A,(HL)
            DEC  A                  ; let gravity work
            LD   (HL),A
            JP   P,sete             ; test ABS
            NEG                     ; dy step positive
sete        LD   E,A                ; set dy step

wind        LD   A,0                ; get wind
dx          ADD  A,0                ; add horizontal move
            LD   D,A                ; total move horizontally

plot        PUSH DE
            PUSH BC
            CALL makescr            ; built the screen, test hit
            POP  BC
            POP  DE

            XOR  A
hit         OR   0                  ; other hit, not own?
            JR   NZ,start           ; hit, in range of cannon
            OR   E
            JR   Z,reach1           ; dy steps done

dy          LD   A,0
            ADD  A,A
            CCF
            SBC  A,A
            CPL
            ADC  A,C
            CP   4
gamelpjr    JR   Z,gameloop         ; bottom screen, missed
            LD   C,A
            DEC  E

reach1      XOR  A
            OR   D
            JR   Z,stepdone         ; dx steps done
            LD   A,B
endx        SUB  4                  ; starts with move to right
            JR   Z,gamelpjr         ; shot over cannon
            SBC  A,A
            CPL
            ADC  A,B
            LD   B,A                ; NEVER zero
            DEC  D

stepdone    LD   HL,frames          ; nice delay for steady screen
            LD   A,(HL)
            SUB  2
wfr         CP   (HL)
            JR   NZ,wfr

            LD   A,D
            OR   E
            JR   NZ,plot            ; either x or y move to do
            JR   dylp               ; full step done, do gravity

makescr     LD   HL,cbdata
            PUSH HL
; "cls" cannonball lines
            LD   (HL),C             ; set y
            LD   A,B
            LD   (cannonx+1),A
```

```
            LD   DE,3
            LD   B,31
clcb        INC  HL
            LD   (HL),D
            DJNZ clcb
            INC  HL
            DEC  C
            LD   (HL),C             ; next y
            DEC  E
            JR   NZ,clcb-2          ; do 3x y
            LD   (HL),E             ; end of ball marker = 0
; plot the cannonball
cannonx     LD   A,0
            AND  #F8
            RRCA
            RRCA
            RRCA                    ; div 8
            LD   B,A
            INC  B
            POP  HL
fstart      INC  HL
            DJNZ fstart
; not with pairs since DE=0
            LD   C,%11100000        ; C with E
            LD   B,%01000000        ; B with D
            LD   A,(cannonx+1)
            AND  7
            JR   Z,shdone
shbit       RR   B
            RR   D
            RR   C
            RR   E
            DEC  A
            JR   NZ,shbit
shdone      LD   (HL),B
            INC  HL
            LD   (HL),D
plotball    LD   A,L
            ADD  A,31
            LD   L,A
            LD   (HL),C
            INC  HL
            LD   (HL),E
            LD   A,B
            CP   C
            LD   C,B
            LD   E,D
            JR   NZ,plotball

            LD   L,cbdata*256/256+1 ; Setting L can save 1
            EXX
            LD   HL,#4000           ; pointers left cannon
            LD   DE,cldata          ; start of left data
cl          LD   A,0                ; height cannon left
            CALL setcannon          ; display in screenmemory

            EXX
            LD   L,cbdata*256/256+32-5 ; right cannon positio
            EXX
cr          LD   A,18               ; height cannon right
            LD   L,#0A              ; pointers right cannon
            LD   E,crdata*256/256   ; start of right data

setcannon   LD   BC,(cbdata)        ; C holds Y of cannonball
```

```
            LD   B,10              ; height of cannon is 10 lines
sc          PUSH AF
            SUB  C
            JR   Z,cbl             ; and stop now
            ADD  A,2
            JR   C,cbl-1           ; skip line, should be placed
            POP  AF                ; but no room for code
            LD   (DE),A            ; otherwise set linepointer
            INC  DE
            PUSH AF
            LD   A,(HL)
            LD   (DE),A
            SCF
            INC  DE
            INC  HL
cbl         CALL NC,cbline
            POP  AF
            DEC  A                 ; next line of cannon
            DJNZ sc
            EX   DE,HL             ; cannon now set
            LD   B,A               ; now fill with rock under

setrock     LD   A,B
            DEC  A
            JR   Z,not1            ; not on line 1, nothing
            INC  A                 ; allowed or hires crashes
            CP   C
            LD   A,rockline*256/256
            JR   Z,callcb
            LD   (HL),B            ; set rockdata
            INC  HL
            LD   (HL),A
            INC  HL
callcb      CALL Z,cbrock
            DJNZ setrock
not1        LD   (HL),A            ; all lines set
            EX   DE,HL
            RET

cbline      LD   A,(HL)            ; fetch data
            INC  HL                ; point to next
cbrock      EXX
            LD   E,A
            LD   D,#43             ; pointer to correct data
            LD   B,5
setcb       LD   A,(DE)            ; fetch set udg
            LD   C,(HL)            ; fetch current screen
            OR   C                 ; OR screen with data
            LD   (HL),A            ; set to sceen
            LD   A,(DE)            ; fetch data again
            XOR  (HL)              ; filter of screenvalue
            SUB  C                 ; compare old screen
            JR   Z,nocol

            LD   A,(dy+1)
            ADD  A,A
            SBC  A,A
            JR   Z,sethit          ; reset on upgoing
            XOR  (HL)              ; wall is no hit
sethit      LD   (hit+1),A         ; set hit range

nocol       INC  HL                ; next screenbyte
            INC  DE                ; next udg data
            DJNZ setcb
```

```
            LD   A,32-5            ; 5 steps done above
            ADD  A,L
            LD   L,A
            EXX
            DEC  C                 ; next y
            CP   cbdata*256/256+96 ; 3 lines done?
            RET  C
            LD   C,A               ; all done
            RET

rnd         LD   DE,0
            LD   HL,(frames)
            ADD  HL,DE
            INC  HL
            LD   A,H
            AND  #1F
            LD   H,A
            LD   (rnd+1),HL
            LD   A,(HL)
frnd        SUB  13
            JR   NC,frnd
            ADD  A,25
            RET

lowres      DEFB 118
            DEFB "A"-27,"N"-27,"G"-27,"L"-27,"E"-27,14
angle       DEFB 28,28
            DEFB 0,"S"-27,"P"-27,"E"-27,"E"-27,"D"-27,14
speed       DEFB 28,28
            DEFB 118

hr          LD   B,14
hr00        DJNZ hr00

            LD   HL,lowres+#8000
            LD   A,#1E
            LD   I,A
            LD   A,L
            LD   A,#F5
            LD   BC,#208           ; effective 16 lines
            CALL #2B5

            LD   B,4
hr01        DJNZ hr01

            LD   A,#43
            LD   I,A               ; highbyte of hiresscreen
            LD   IY,lowiy          ; return from highmemory
            LD   IX,lowix          ; return from highmemory
            LD   HL,cldata         ; cannon left data
            LD   DE,cbdata         ; cannonball data
            LD   B,193-16          ; nr of lines+1 - 2 lowres
            EXX
            LD   (hlasv+1),HL      ; save HL' can be used in main
            LD   HL,crdata-1       ; cannon right data

lowloop     NOP                    ; timing only
            INC  HL                ; adjustment for cannonright
            EXX                    ; back to main
lowiy       DEC  B                 ; do next line on screen

lineloop    LD   A,B               ; fetch line number
            CP   (HL)              ; check cannon left
            JP   Z,lcancopy        ; handle leftcannon
```

```
            LD   A,(DE)             ; swap with cannonball
            CP   B                  ; check cannonball
            LD   A,E                ; preload data-1
            LD   R,A                ; preload data display
            JP   Z,ballbuf+#8000    ; directly to display with

            PUSH HL
            EX   (SP),HL            ; syncdelay for rightcannon
            POP  HL

lowix       LD   A,B                ; return leftcannon display
            EXX                     ; swap for test on rightcannon
            CP   (HL)
            INC  HL                 ; point to data line
            LD   A,(HL)             ; fetch data line
            JP   Z,rcanbuf+#8000    ; directly to highmemory

            DEC  HL                 ; undo false increment
            EXX                     ; back to main

            EX   (SP),HL            ; synch line timing
            EX   (SP),HL
            RET  Z
            NOP
            NOP
            DJNZ lineloop           ; only exit from HR-loop

            EXX
hlasv       LD   HL,0               ; HL' back
            EXX

            LD   IY,#4000
            CALL #292               ; and back to program
            CALL #220
            LD   IX,hr
            JP   #2A4

; table of sinus and cosinus set in ballbuffer.
; values multiplied by 64, angle step 5 degrees.
sin         DEFB 1,6,11,17,22,27
ballbuf     DEFB 32,37,41,45,49,52
            DEFB 55,58,60,61,62,63
cos         DEFB 63,63,62,61,60,58
            DEFB 55,52,49,45,41,37
            DEFB 32,27,22,17,11,6,1
            ADD  A,32               ; 48K bug
            LD   E,A                ; point to next line
            RET  Z                  ; timing only
            JP   (IY)               ; back to low from cannonball

lcancopy    EQU  #4014
cldata      DEFB 0                  ; prevent start display
            LD   HL,lcpointer       ; further unpacking
            LD   DE,#4000           ; of data copy over
            LD   BC,10              ; sysvar of pointers
            LDIR                    ; then memory reuse
            LD   HL,crdata+1        ; for screenlines
            LD   C,10
            LDIR
            LD   C,6
            LD   HL,lcan            ; also over sysvar
            LDIR
            LD   HL,code1           ; code on 'screen'
            LD   DE,code            ; copied over sysvar
```

```
           LD   C,11              ; saves 3 bytes
           LDIR
           JP   start

lcpointer  DEFB lc1*256/256       ; cannon left data
           DEFB lc2*256/256
           DEFB lc3*256/256
           DEFB lc4*256/256
           DEFB lc5*256/256
           DEFB lc6*256/256
           DEFB lc7*256/256
           DEFB lc8*256/256
           DEFB lc9*256/256
           DEFB lc10*256/256

fill2      EQU  cldata+51-$       ; total 51 bytes for left
           DEFS fill2

crdata     DEFB 0,rch             ; cannon right data
           DEFB rch+4
           DEFB rch+8
           DEFB rch+12
           DEFB rch+16
           DEFB rch+20
           DEFB rch+24
           DEFB rch+28
           DEFB rch+32
           DEFB rch+36

packcr     DEFB 56,0,108,0,70,0,111,132
           DEFB 63,236,15,248,3,190,1,95
           DEFB 1,191,0,238,0,0,0,0

lcan       INC  L                 ; point to data line
           LD   A,(HL)            ; fetch data line
           INC  L                 ; point to next linenumber
           JP   lcanbuf+#8000

filler     EQU  crdata+51-$

           DEFS filler            ; total 51 bytes for right

SPACE      EQU  #4318-$           ; codeable memory, rest screen

           DEFS SPACE             ; free codeable memory left
; no byte left in this version

lcanbuf    LD   R,A
lc1        DEFB 0,0,0,28,0        ; first left cannonline
           RET  C                 ; timing
           RET  C                 ; timing
           JP   lowix

lc3        DEFB 0,0,0,98          ; 3rd line
lc4        DEFB 0,0,33,246        ; etc...
lc5        DEFB 0,0,55,252
lc6        DEFB 0,0,31,240
lc7        DEFB 0,0,125,192
lc8        DEFB 0,0,250,128
lc9        DEFB 0,0,253,128
lc10       DEFB 0,0,119

basic      DEFB 0
rc1        DEFB 0                 ; first right cannon line
```

```
rch         EQU  rc1*256/256       ; rest is built from other mem

            DEFW 0                 ; then data for right cannon
            DEFB 249,212,28
            DEFB 126
            DEFB 143,0,18,0,0

dfile       DEFB 118               ; full lowres screen for init
            DEFB 118,118,118,118,118,118,118,118
            DEFB 118,118,118,118,118,118,118,118
            DEFB 118,118,118,118,118,118,118,118

fill3       EQU  rc1+42-$

            DEFS fill3
; memory filler for right cannon

rockline    DEFB 255,255,255,255,255

rcanbuf     RET  NZ                ; timing
            BIT  0,(HL)            ; timing
            LD   R,A
lc2         DEFB 0,0,0,54,0
            JP   lowloop           ; 48K bug

cbdata      DEFB 200               ; on start out of screen
init        LD   SP,#4400          ; in game 3 lines of cannonbal
            LD   HL,#4000
            LD   DE,#C000
            LD   BC,#400
            LDIR                   ; repair 48K bug

            LD   DE,packcr         ; packed cannon right
            LD   HL,rc1+1          ; unpack right cannon
            LD   C,10
sl0         LD   B,2
sl          LD   A,(DE)
            LD   (HL),A
            XOR  A
            INC  HL
            INC  DE
            DJNZ sl
            LD   (HL),A
            INC  HL
            LD   (HL),A
            INC  HL
            DEC  C
            JR   NZ,sl0
            LD   IX,hr             ; hires mode
            JP   cldata            ; set table and start

code1       ADD  HL,HL             ; 11 bytes to copy over
            SBC  A,A               ; sysvar
            JR   c2
            DEFB 255,255,255       ; not codeable
            DEFB 55                ; not codeable
c2          RET  C
            LD   A,H
            RET

vars        DEFB 128
?
last        EQU  $
```