
Mehrzweckhalle

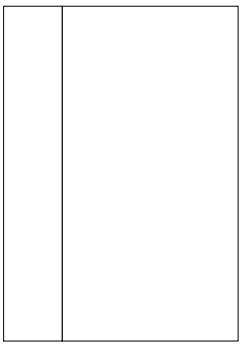

Herzlich Willkommen
in der JVA Bochum

o1 - KONZEPT // HERLEITUNG

Die Wahl des Konzepts wurde durch eine Arbeit in meinem Praktikum beeinflusst. In Reiden wurde kürzlich die neue Mehrzweckhalle eröffnet. Die 3-Fach-Halle überzeugt unteranderem durch eine aussergewöhnlich ausgeklügelte Decken organisation. Oblichter, verschiedene Leuchtkörper [Röhren und Spots], Rauchabzug, Rauchmelder, Beschilderung, Ringe, etc. sind das einfach Plattenmuster integriert. Dabei scheinen die einzelnen Elemente auf den ersten Blick frei versetzt, was in Wahrheit nicht so ist. Obwohl die gesammte Decke “manuell” gestaltet wurde, bin ich der Auffassung, dass man dies auch programmieren könnte. Ich wollte versuchen, quasi mit einem Programm den alltäglichen Anordnungen an den Decken ein Ende zu bereiten, aufzuzeigen, wie auch anders geht. Ein hochgestecktes Ziel, welches ich leider nicht wirklich einhalten konnte.

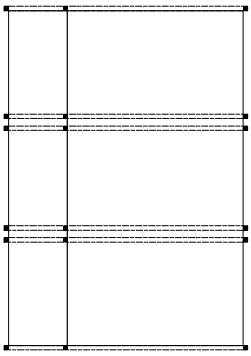
Da ich keinerelei Vorkenntnisse betreffend Programmieren besitze, habe zwar eine vage Vorstellung, aber auch nicht mehr, wie ein solchen Programm aussehen könnte, wie es aufgebaut ist und wie es funktioniert. Welche Erfahrungen und Erkenntisse ich gesammelt habe und auf welche Irrwege und Probleme ich gestossen bin werden in den laufenden paar Seiten dargestellt.

o1 - KONZEPT // PROGRAMM



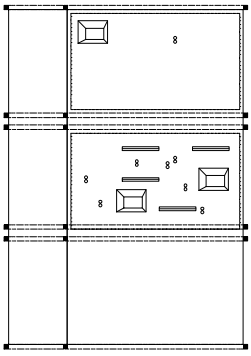
FAKTOREN

Die Art der Halle
> Hallengrösse
> Unterteilungen
>> Unterzüge (3-Fach-Halle)
>> Tribüne
Konstruktion:
> Stützen
> Unterzüge
Gleichmässige Ausleuchtung:
> Oblichter
> künstliche Beleuchtungskörper



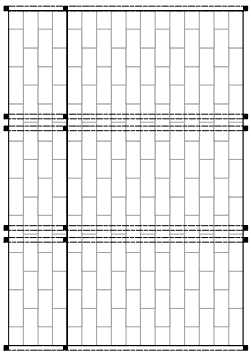
SPIEL

Das Konstruktionsraster, respektive das Plattenraster der Decke gibt den Perimeter vor, in welchem sich diverse Elemente nach irgendwelchen gesetzen umher bewegen.



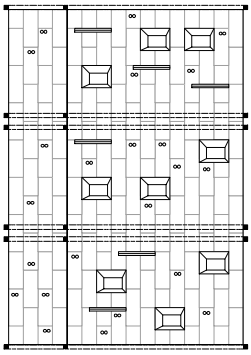
SPIELREGELN ELEMENTE

bsp. Oblichter stossen sich ab
bsp. treffen 2 zusammen, gibt es eine neue Leuchte
bsp. Konstruktionsraster passt sich an die Bewegungen an
etc.



SPIELREGELN USER

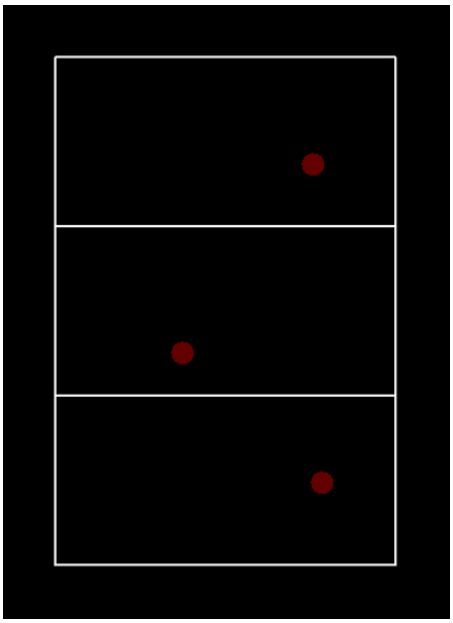
bsp. in jedem Perimeter gibt es 3 Oblichter
bsp. Anzahl Spots
bsp. Hallengrösse und Unterteilung
etc.



ZIELE

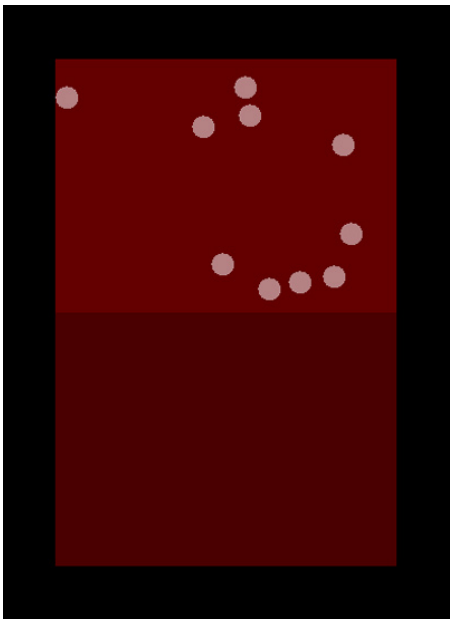
> Deckengestaltung der etwas anderen Art
> Rasche Variation
> Selbstregulierende Deckenplanung
> Ausgangslage / Regeln / Resultat

o2 - ERSTE SCHRITTE



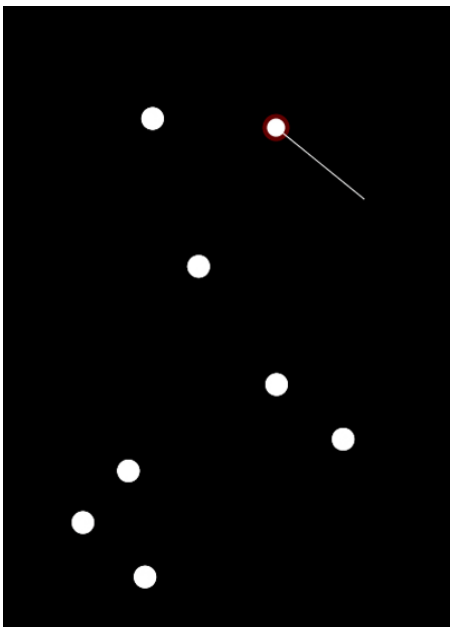
VORGEGEBENER AKTIONSBEREICH - TEIL 1

Das Programm zeichnet eine Aussengrenze sowie innere Grenzen. Denn Bällen ist es verboten, diese Grenze zu überschreiten.



VORGEGEBNER AKTIONBEREICH - TEIL 2

Als weiterer Versuch besitzen die unterschiedlichen Parameter verschiedene Farben. Die Bälle haben hier den Befehl, sobald der Hintergrund nicht mehr Hellrot ist, dann kehre sofort um.
Der Ball testet jeweils welche Farbe unter ihm liegt. Solange sie sich auf der Hellroten Fläche befinden schweben sie frisch fröhlich vor sich hin.

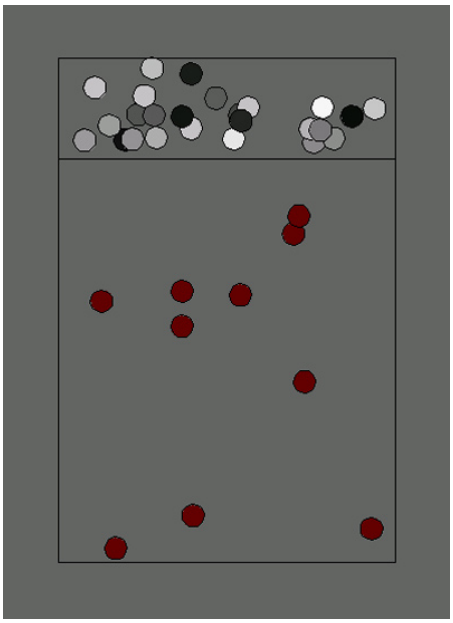


BEWEGUNG DER BÄLLE UND ERWEITERUNG

Die Bewegung der Bälle ist sehr einfach. Allerdings bereitet das richtige Abprallen an der Grenze einige Schwierigkeiten. Das Problem ist, dass der Ball nicht weiss wie, er an eine Grenze stösst. Ob er an die obere, untere, rechte oder linke Grenze geraten ist aber wesentlich für seine weitere Flugrichtung.

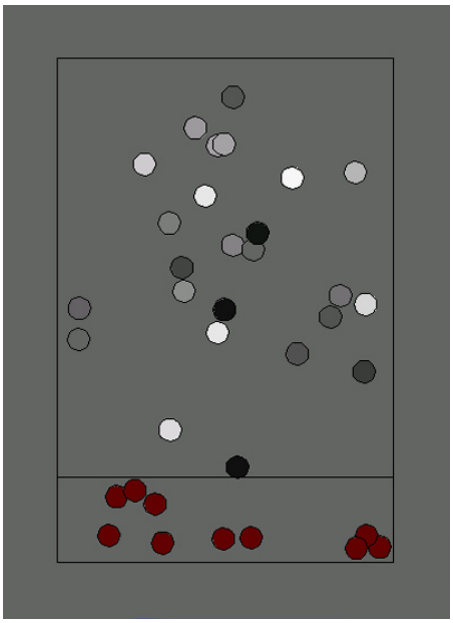
Der Ball der am nächsten zur Maus ist wird markiert. Der Ball wird mit einer Linie mit der Maus verbunden. In einem weiteren Schritt kann so der nächste Ball per Mausklick gelöscht werden.

o3 - ERSTES RESULTATE



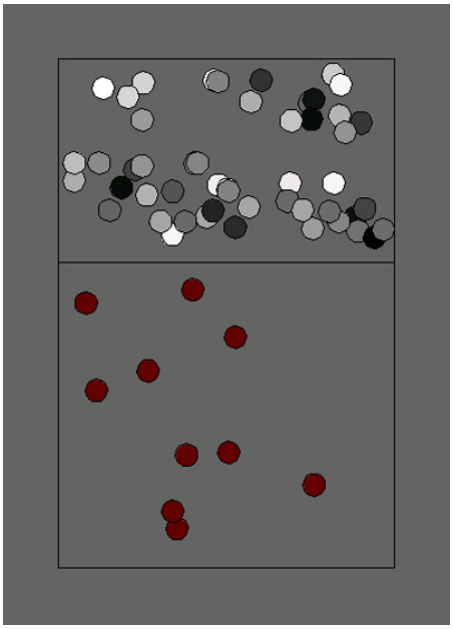
ERSTES SPIEL

In diesem Programm werden zweimal die selben Bälle gezeichnet. Es werden ihnen jedoch andere Grenzbereiche und Startpositionen, sowie eine andere Farbe zugewiesen. Die Bälle schweben in ihrem Bereich umher und kehren um, sobald sie an die Grenzen stossen.
Es können nicht einfach die Mittelpunkte der Bälle betrachtet werden, da der einzelne Ball scho [Radius = r] vor der Grenze umkehren soll.



MOBILE GRENZE

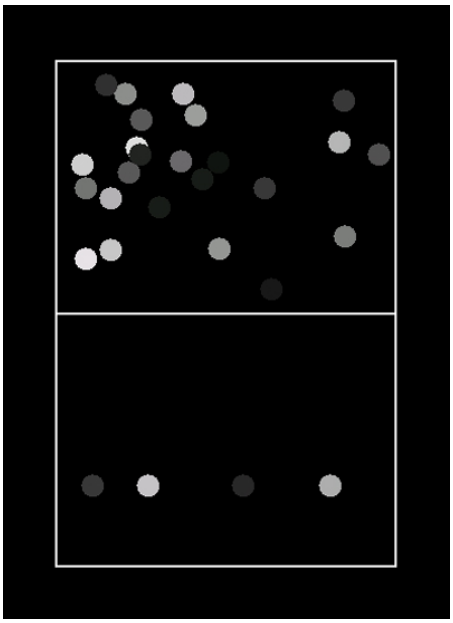
Die Grenze bzw. die Line in der Mitte ist jeweils auf der Höhe wie der Mauszeiger. Sie besitzt aber einen Min und einen Max Wert, damit sie immer innerhalb des Rahmens bleibt. Per Knopfdruck lässt sich die Grenze nach oben, respektive nach unten verschieben. Bei jedem verschieben der Linie, wird auch der Grenzbereich angepasst. Der Ball muss wissen, dass die Grenze verschoben worden ist.



ERSTES SPIEL

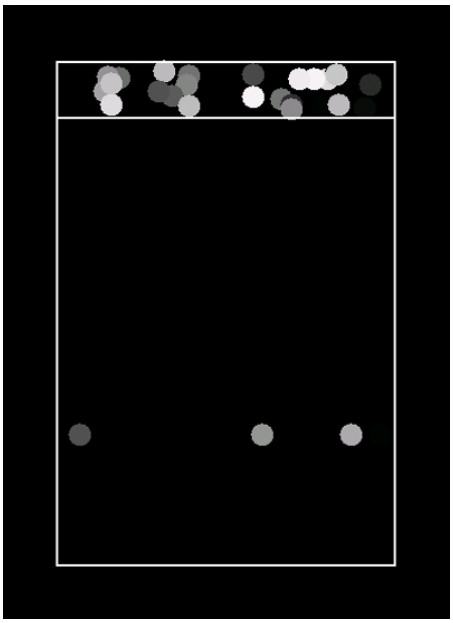
Ein Problem das ich bis heute noch nicht lösen konnte entsteht, wenn sich die Grenze auf den Ball zubewegt und dieser sich selbst in Richtung der Grenze bewegt. Dann kann es sein, dass der Ball bereis über der Grenze ist, wenn er das nächste Mal seine Position überprüft. Dann kehrt er zwar sofort seine Richtung, befindet sich aber beim nächsten Durchlauf immernoch im verbotenen Bereich und wechselt deshalb die Richtung nochmal, und nochmal, und nochmal.. Als Resultat bleibt der Ball hängen. Er flimmert.
Daraus ist aber ein Spiel geworden: Man bringen alle Bälle drucheinander und versuche dann mit der Maus, respektive mit der Linie alle Bälle wieder in ihren Bereich zu bringen.

o3 - ERSTE RESULTATE



ERSTES SPIEL - VERSUCHE

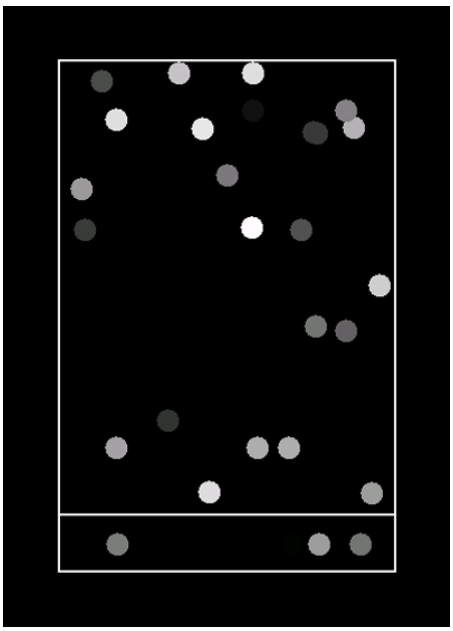
Dieses Programm gleicht dem vorherigen. Allerdings bleiben hier die Bälle nicht mehr hängen. Dieses Problem konnte ich umgehen, jedoch nicht ohne ein neues zu schaffen. Ich habe dem Ball jeweils befohlen die Richtung für längere Zeit nicht mehr zu änderen, direkt nachdem er eine Grenze erreicht hat. Allerdings stimmen dann die Winkel entweder oben und unten oder links und rechts. An allen vier Seiten allerdings leider nie.



MOBILE GRENZE

Die Grenze bzw. die Line in der Mitte lässt sich hier mit Tasten nach oben respektive unten verschieben. Wiederum wurde ein Minimum und ein Maximum definiert.

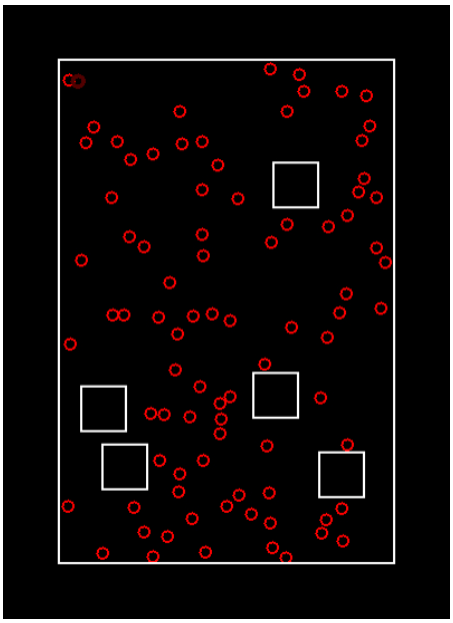
“w” = Grenze wird um 1 nach oben verschoben
“s” = Grenze wird um 1 nach unten vershoben



VARIATION

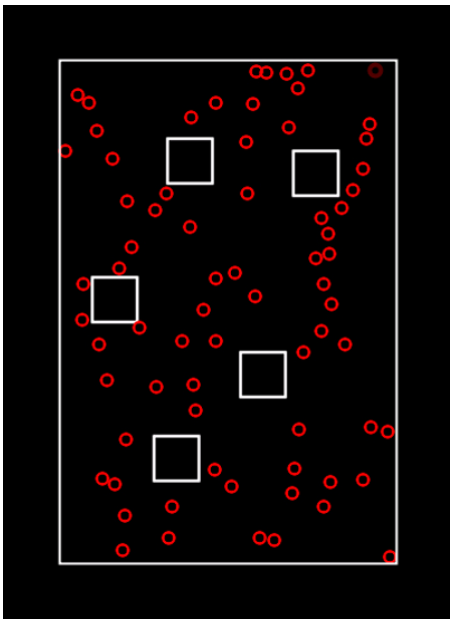
Wie viele Bälle in den jeweilgen Bereichen umherschweben gibt man beim Programmstart an.

o3 - ERSTE RESULTATE



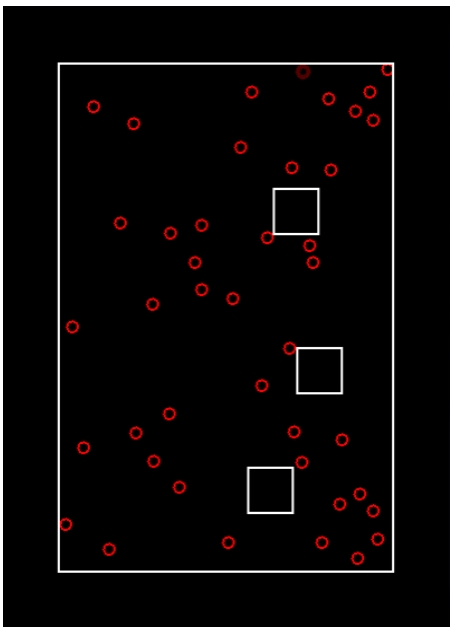
DECKENORGANISATION_o1

Die Leuchten [rote Kreise] und die Oblichter [kleine, weisse Vierecke] schweben im vorgegebenen Perimeter umher. Die Bälle wechseln ihren Winkel, sobald sie auf “weiss” stossen. Jeder Ball überprüft deshalb immer, ober er unter sich weiss hat.
Die Oblichter tun das selbe. Für beide Elemente gibt man für den Start eine Anzahl an. Lechten als auch Oblichter bewegen sich und werden immer langsamer, da ich eine “Bremse” eingebaut haben. Solange, bis alles stillsteht.



VARIATION

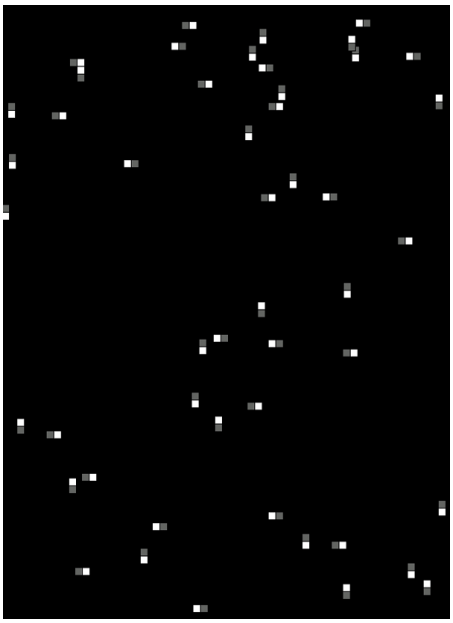
Bei jedem Knopfdruck entsteht ein anderes Deckenmuster. Das Programm kann noch lange nicht, was es schlussendlich alles können muss, aber immerhin es funktionert ein klein wenig.



ERKENNTNISSE

Die Bälle prallen bei den Oblichtern nicht immer richtig ab. Das Problem ist, dass der Winkel der Flugrichtung sich anders ändern muss, wenn der Ball unten an eine Grenze stösst, als wenn er rechts in eine Grenze stösst. Diese Tatsache hat mich lange Zeit verfolgt. Der Ball weiss nicht, wie er an eine Grenze gelangt. Er weiss nur, oha: weiss! umkehren!
Aber genau dieses Umkerhren, macht er nicht richtig.

o4 - TURNHALLE START

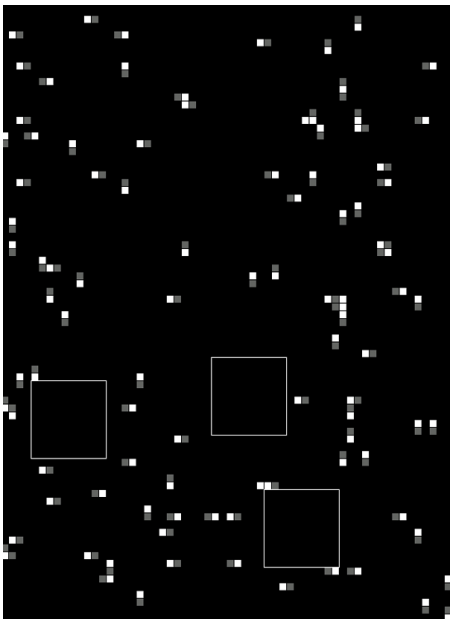


NEUER ANSATZ

Um das Programm und die Bewegungen der einzelnen Elemente in den Griff zu bekommen, habe ich einen anderen Ansatz gewählt. Zentrales Element dieses Programmes ist das Array im Hintergrund, welches speichert, was genau sich auf jedem einzelnen Array befindet.
Die Leuchten bewegen sich zufällig nach oben, unten, links und rechts.

Im ersten Schritt sind dies:
1 = linker Rand: Alle Leuchten dürfen nicht nach links
2= oberer Rand: Alle Leuchten dürfen nicht nach oben
3 = rechter Rand: Alle Leuchten dürfen nciht nach rechts
4 = unterer Rand: Alle Leuchten dürfen ncicht nach unten

Dieser Ansatz hatte so aber keine Zukunft.

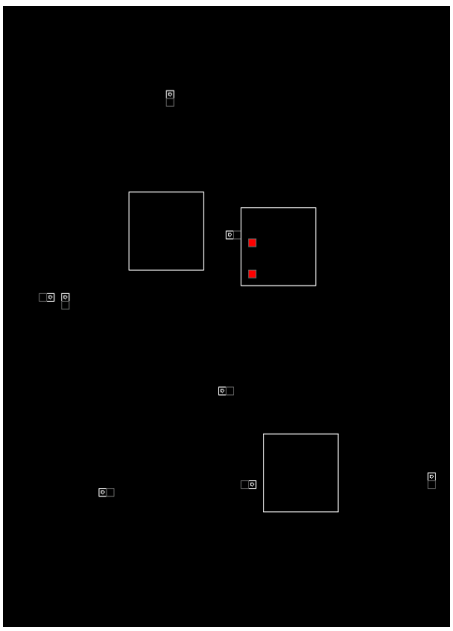


NEUER ANSATZ - ABER BESSER

Naja, das Bild sieht genau gleich aus. Trotzdem hat sich hier grundlegend etwas geändert. Die Leuchten sind nun viel “intelligenter”. Sie überprüfen jeweils bevor sie sich verschieben, ob das Feld, welches sie besetzen möchten überhaupt noch frei ist. Das Array hat nun folgende Werte gespeichert:

0 = freies Feld: Hier darf ich hin
8 = Oblicht: Hier darfst ich nicht hin
9 = Grenze: Hier darf ich nicht hin

Für die Oblichter gelten genau die selben Regeln. Als Kontrollpunkte werden alle 4 Ecken überprüft. Respektive die beiden Ecken, welche in der Bewegungsrichtung liegen. Erst wenn diese die Bedingung [frei] erfüllen, bewegen sie sich.

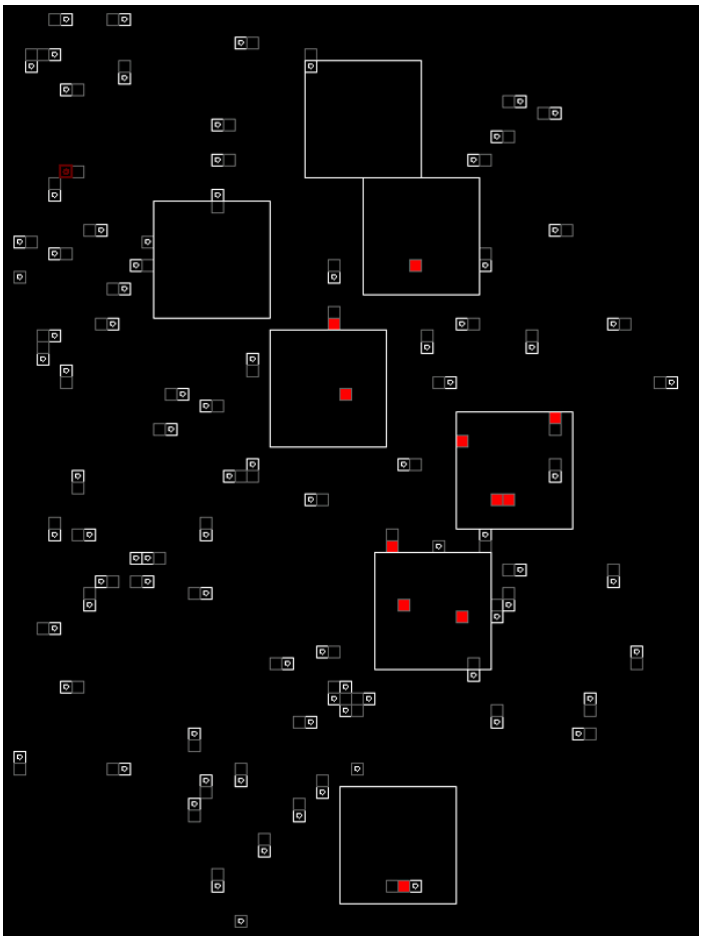


OBLICHTER FRESSEN LEUCHTEN

Den Oblichter wird in einem weiteren Schritt erlaubt, die Leuchten zu “fressen”. Später soll dann mit diesen “gefressenen” Leuchten irgendwas passieren. Vorerst aber habe ich mich damit begnügt, dass die Leuchte erkennt, das sie gefressen wurde. Und wenn sie gefressen ist, dann soll sie rot werden.

Das Oblicht schreibt bei jeder Bewegung alles was hinter sich liegt mit dem Wert [Oblicht = 8]. Was man natürlich bedenken muss, ist dass der Ort wo sich die Elemente vorher befand wieder mit dem Wert [Frei = 0] beschrieben wird.

o4 - TURNHALLE



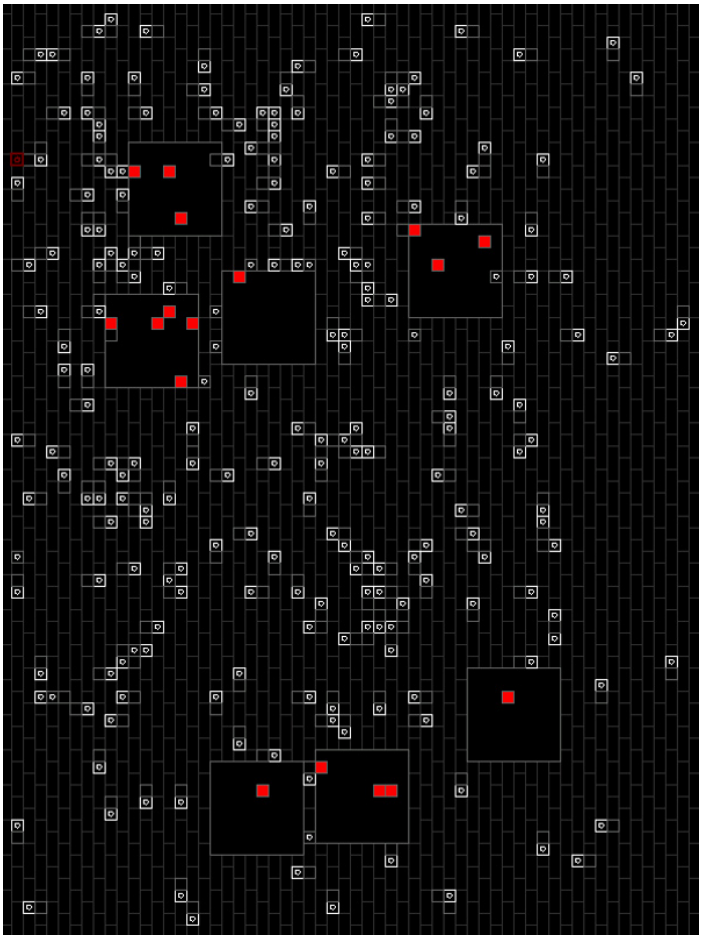
ERWEITERUNGEN

Die Erkenntnisse der früheren Versuche habe ich dann in diesem Projekt eingeführt.
Per Mausklick können Leuchten sowohl erstellt als auch gelöscht werden.

Mausklick Links:
An der Position der Maus wird eine neue Leuchte gezeichnet.

Mausklick Rechts:
Die Leuchte, die sich am nächsten an der Mausposition befindet wird farbig markiert. Drückt man die rechte Maustaste, so wird diese markierte Leuchte gelöscht.

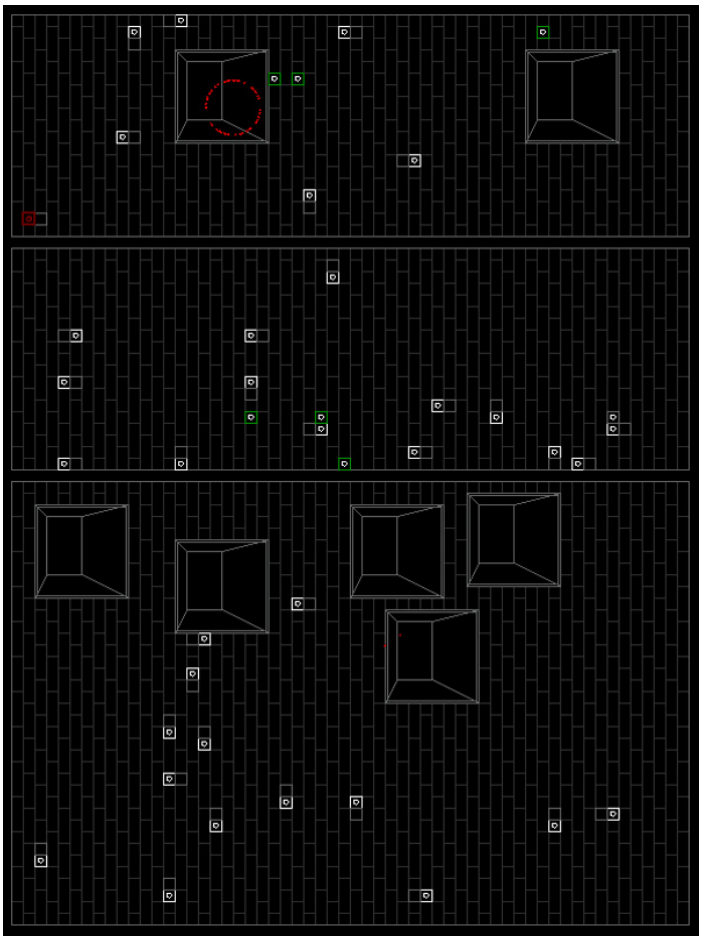
Später sollen auch Oblichter und alle anderen Elemente per Maus oder doch zumindest per Tastendruck erstellt und gelöscht werden.



VARIATION

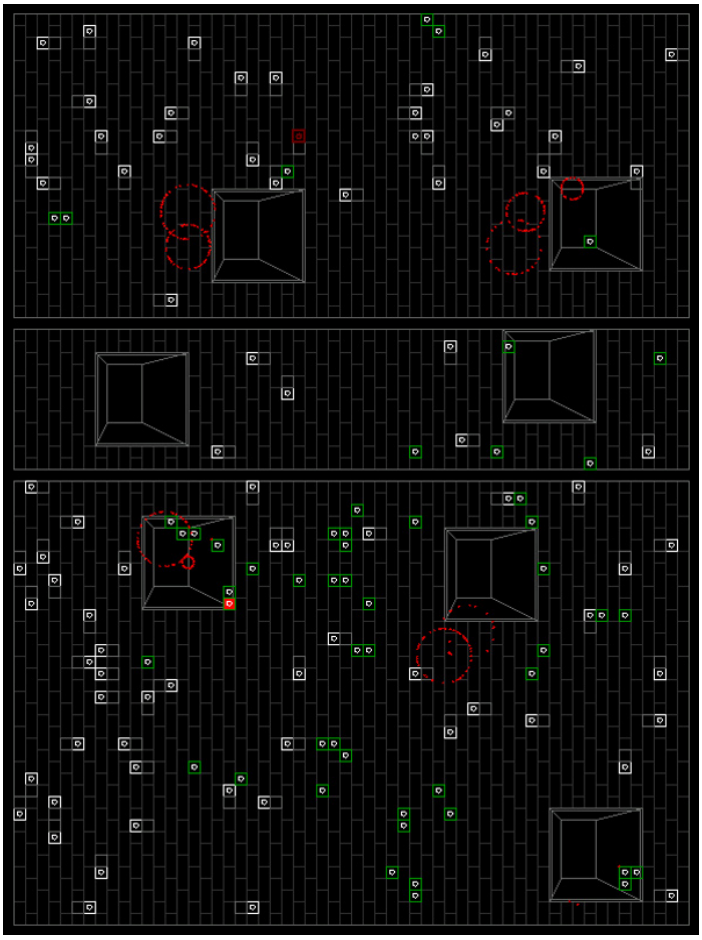
Die Leuchten und die Oblichter sind jeweils eigene Klassen. Deshalb lässt sich die Anzahl beliebig durch Angaben beim Start variieren.
Per Maus und Tastendruck lassen sich die Elemente im Nachhinein vermehren und vermindern.

o4 - TURNHALLE KONSTRUKTION



UNTERZÜGE

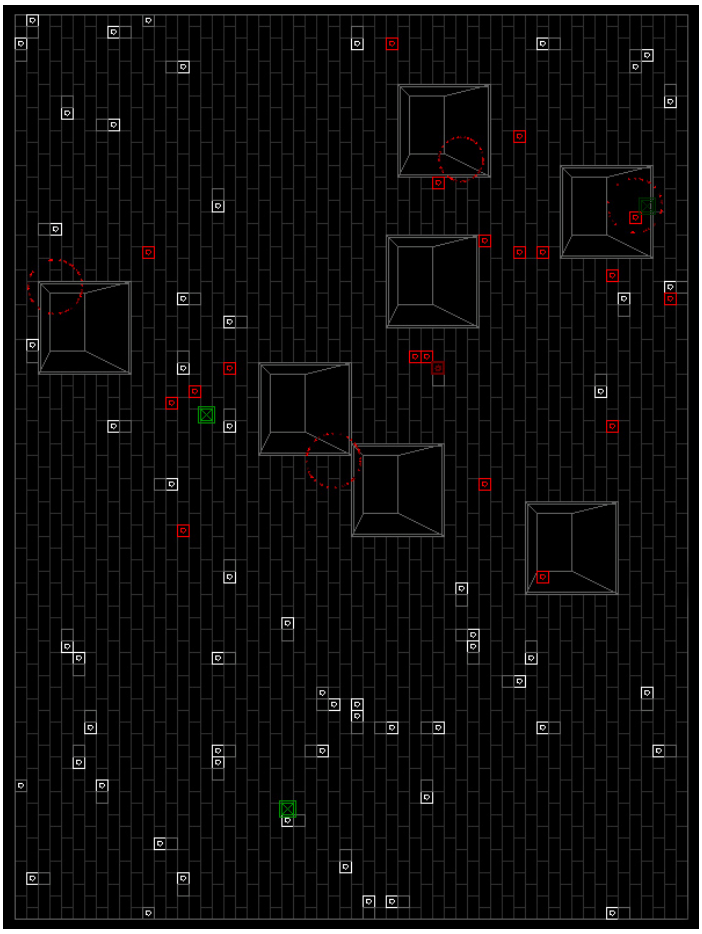
Bisher habe ich das Ausgabefenster jeweils als ein Perimeter betrachtet. Zuerst müssen die Abläufe in einem Feld richtig funktionieren, bevor ich den Aktionsbereich durch Unterzüge in Verschiede Bereiche unterteile.
In dieser Version kann man die Anzahl der gewünschten Unterzüge angeben. Das Programm zeichnet mir die Unterzüge mit der Mindestbreite von einem ArrayFeld. Das Plattenraster wird überzeichnet
> visuelle Grenze
Im Array wird für die Unterzüge den Wert [Grenze = 9] gesetzt. Oblichter sowohl Leuchten sind in ihrem eigenen Perimeter gefangen.



UNTERZÜGE BEWEGEN

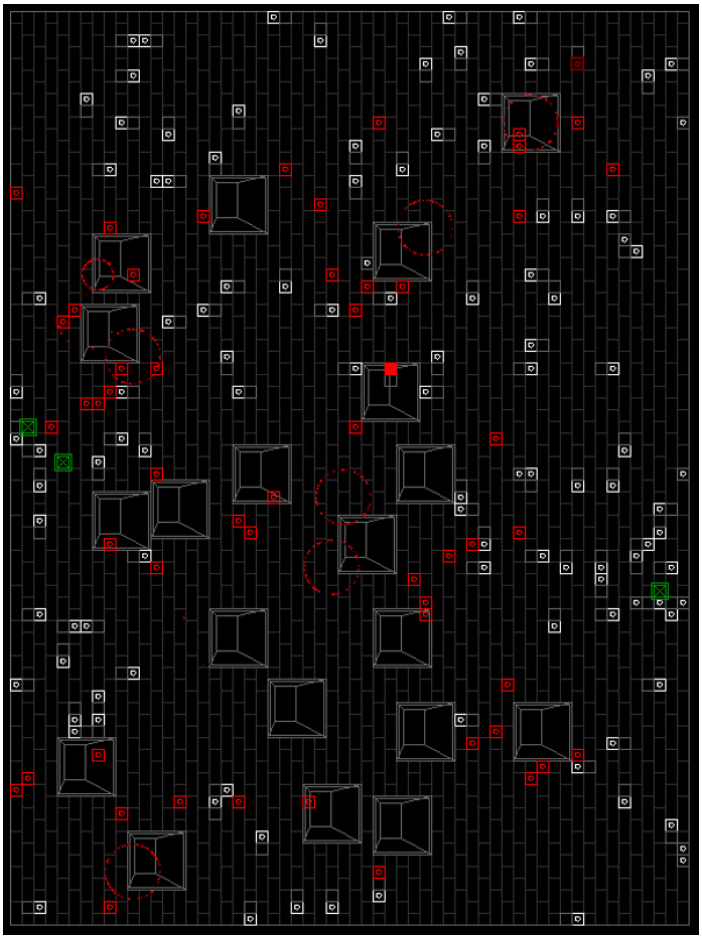
Per Tastendruck kann man die Träger im Nachhinhein verschieben. Die Oblichter und Leuchten werden gezwungen sich zu verschieben. Sie werden quasi durch die Grenze geschoben. Alle Elemente die auf die Grenze zu liegen kommen, bsp. die FixLeuchten, Explodieren.
Irgendwie hat mich aber die Flexibilität der Unterzüge und das Handling nicht überzeugt, weshalb sie in den nächsten Schritten nicht auftauchen werden.

o4 - TURNHALLE ABGABE



TURNHALLE_o2

Die “Leuchte” (weisses, kleines Viereck) bewegt sich auf dem vorgegeben Raster. Sie können nicht hinaus und sie kann nicht dahin wo bereits ein Oblicht oder eine “FixLeuchte” ist. Wird eine Leuchte von einem Oblicht überfahren, so wird si gelöscht, an ihrere Stelle entsteht eine “Fixleuchte”.
Die “Fixleuchte” (rotes, kleines Viereck) kann steht im Prinizip nur rum. Sie überprüft aber jeweils, ob sie in einem Oblicht steht. Hällt sich die Leuchte zu lange im Oblicht auf, so Explodiert sie. D.h. Anzahl i Bälle werden gezeichnet, die von ihrem Ursprung davon fliegen und nach j durchläufen, verschwinden alle Punkte wieder.
Die “Oblichter” (grosses, weisses Viereck) kann nicht über die Grenzen fahren und es kann nicht auf ein anderes Oblicht. Dafür kann sie “Leuchten” “fressen” und “Fixleuchten” zum explodieren bringen.
Der “Elektriker” (grünes, kleines Viereck) erstellt nach k durchläufe eine neue “Leuchte”. So läuft dieses Spiel immer weiter. Fressen und gefressen werden!



ERWEITERUNG

Jenste Knöpfe sind Programmiert, welche Elemente entstehen lassen oder aber löschen.

“Maus links” = neue Leuchte an Mausposition
“Maus rechts” = löscht Leuchte die am nächsten ist
“w” = neues Oblicht an Mausposition
“s” = löscht Oblicht das am nächsten ist
“e” = neue Fixleuchte an Mausposition
“d” = löscht Fixleuchte die am nächsten ist
“r” = neuer Elektriker an Mausposition
“f” = löscht Elektriker der am nächsten ist

o5 - GEDANKEN ZUM ABSCHLUSS

WIE GEHT ES WEITER?

> Oblichter und Elextriker sterben nicht von selbst. Die Spielregeln sollten so erweitert werden, dass irgendwelche Bedingungen Oblichter und Elextriker neu erschaffen oder umbringen.

> Neue Elemete einfügen ist sehr einfach. Die neue Klasse wird einer der bestehenden sehr ähnlich sehen. Vielleicht gibt es einen weiteren Lichtquellen Typ, bsp. eine Röhre

> Output im 3D. In der ersten Besprechung wurde dies diskutiert. Das Programm funktioniert im 2D einwandfrei. Zugegeben, es ist nicht sehr komplex. Trotzdem eine Umsetzung ins 3D würde eine bessere räumliche Vorstellung vermiteln. Die Umsetzung wird nicht schwierig sein. Es fehlt nur jeweils die 3te Dimension.