

Depon.Net 2.0

Inhalt

Einleitung.....	2
Konzepte.....	3
Flache Welt (2D)	3
Beschreibung.....	3
Grafische Darstellung	3
Spielbarkeit.....	3
Möglichkeiten für den Nutzer	4
Umsetzbarkeit	4
Höhenwelt (2,5D)	5
Grafische Darstellung	7
Spielbarkeit.....	8
Möglichkeiten für den Nutzer	8
Umsetzbarkeit	8
Volumenwelt (3D)	9
Grafische Darstellung	10
Spielbarkeit.....	11
Möglichkeiten für den Nutzer	11
Umsetzung.....	11
Migrationsszenario.....	12

Einleitung

Dieses Dokument beschreibt die möglichen Konzepte, wie die Karte aufgebaut werden kann. Hierbei sollen Vor- und Nachteile für die Bereiche:

- Grafische Darstellung: Wie soll die Karte dem Nutzer präsentiert werden. Dieser Punkt interagiert mit den Punkten 'Spielbarkeit' und 'Umsetzbarkeit'.
- Spielbarkeit: Ist diese Umsetzung für den Spieler verständlich und ohne größere Hürden spielbar? Wird hier eine zusätzliche Komplexitätsebene eingezogen, die keinen spielerischen Mehrwert bietet?
- Möglichkeiten für den Nutzer: Ist mit dieser Umsetzung nur eine sehr eingeschränkte Spielweise möglich oder bietet diese neben den eigentlichen Spielregeln auch Möglichkeiten zur kreativen Auslebung?
- Umsetzbarkeit: Wie sieht es mit der technischen Umsetzbarkeit aus? Hier sind sowohl Schwierigkeiten auf Client-, als auch auf Serverseite zu beachten. Wie ist der Speicherbedarf?

Konzepte

Flache Welt (2D)

Beschreibung

In diesem Konzept ist die Welt als flaches Modell zu betrachten. Jedes Objekt und jedes Feld besitzt X-Y-Werte, die die euklidischen Koordinaten beschreiben.

Jede Koordinate ist einem bestimmten Feldtyp zugeordnet. Man kann hier überlegen, ob es diskrete Felder gibt oder die Zugehörigkeit der Feldtypen über Bedingungen formuliert wird.

Bekannte Spiele wie Civilization und Colonization benutzen dieses Koordinatensystem.

Grafische Darstellung

Die grafische Darstellung lehnt sich an das alte Depon.Net, an Civilization und eigentlich an jedes alte Spiel an.

Vereinfacht gesagt würde es ähnlich dem folgenden Bild aussehen:



Spielbarkeit

Die Spielbarkeit ist gegeben, da der Spieler gewohnt ist zweidimensionale Strukturen in Spielen, insbesondere Strategiespielen zu erfassen.

Die Selektion der Felder erfolgt über das Anklicken der jeweiligen Koordinate. Auch lässt sich so einfach ein Gebäude, eine Einheit und andere Objekte auf der Karte auswählen.

Es besteht keine Gefahr, dass der Nutzer das falsche Element anklickt.

Möglichkeiten für den Nutzer

Dieser Kartentyp lässt eine Manipulation des Feldtypes und damit des Feldaussehens zu.

Der Benutzer ist nicht in der Lage Formen über dreidimensionale Koordinaten zu erzeugen.

Strategische Tiefe ist auch nur über Feldtypen zu erlangen. Das Erklimmen von Bergen oder umgehen von Tälern erscheint immer etwas künstlich

Umsetzbarkeit

Die Umsetzung kann sowohl auf Client-, als auch auf Serverseite mit einfachen Mitteln erzeugt werden.

Im Großen und Ganzen ist dies nur ein geordnetes Zusammenfügen und gegebenenfalls Überschreiben von zweidimensionalen Texturen. Es ist maximal eine Skalierung je nach Zoomstufe notwendig.

Der Speicherbedarf beträgt: $X \cdot Y \cdot \text{Speicherbedarf des Feldtyp}$ Bytes. Möchte man 256 Feldtypen unterstützen, so benötigt eine Karte 1 MB Speicherbedarf. Es steht ein Server mit 8 GB RAM zur Verfügung. Daher sind Karten bis 2-3 GB ohne Einschränkungen möglich. Dies entspricht einer Kartengröße von 50.000 zu 50.000 Feldern. Ist jedes Feld 10 m groß, so ist ein virtuelles Gebiet von 500 km abgedeckt.

Höhenwelt (2,5D)

In der Höhenwelt besitzt jedes Objekt und jedes Feld, wie in der flachen Welt, einen eindeutigen X-Y-Wert in einem euklidischen System. Zusätzlich besitzt jedes Objekt und jede Feldkoordinate einen Höhenwert.

Weiterhin ist zu jeder Feldkoordinate ein bestimmter Feldtyp zugeordnet.

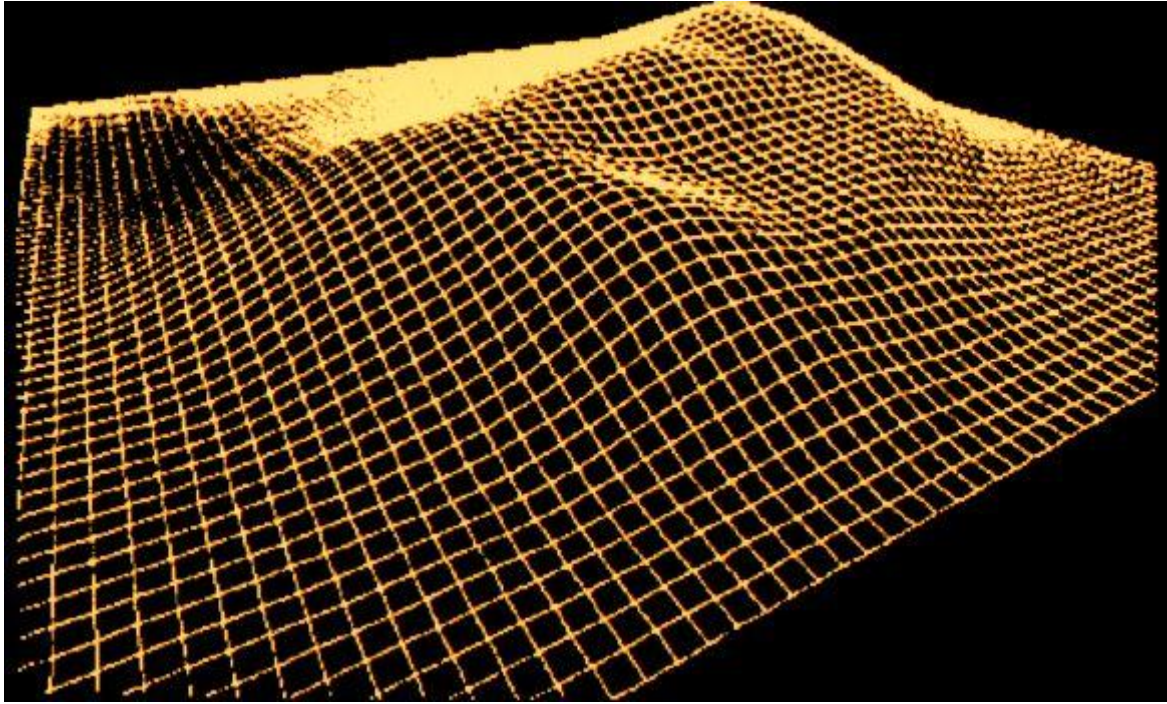
Der Höhenwert bestimmt die Höhe des Objektes in einem dreidimensionalen Körper. Es ist nicht möglich, dass zu einer X-Y Koordinate mehrere Felder gefunden werden.

Die Welt ist damit nicht überdeckend und zu jeder X-Y-Koordinate wird nur ein Feldtyp zu einer Höhe gefunden.

Beispielsweise kann hier das Spiel 'Siedler 2' genannt werden:



Natürlich lässt sich eine solche Grafik nur mit professioneller Grafikgestaltung umsetzen, die mir nicht zur Verfügung steht. Daher folgende weitere Bilder, die eine Idee der Darstellung zeigen.





Grafische Darstellung

Die grafische Darstellung kann über eine isometrische Darstellung erfolgen der Karte selbst, bei der die Objekte nachträglich positioniert werden können.

Das Rendering kann sowohl auf dem Server, als auch auf dem Client durchgeführt werden. Es empfiehlt sich allerdings die Datenlast zwischen Server und Client gering zu halten, so dass ein Rendering beim Client vorgezogen werden sollte.

Neben Skalierung und Translation ist bei dieser isometrischen Darstellung auch eine Rotation und eine Scherung der Elemente erforderlich. Dies ist nicht mehr mit Mitteln, die einem HTML 4-Browser¹ zur Verfügung stehen.

HTML 5 ist für ein vollständiges Rendering auf 1920x1080 Pixel (Größe vieler Monitor) noch zu langsam. Dies ist allerdings zu testen. Dazu ist eine Implementierung mit Canvas und mit SVG umzusetzen.

WebGL ist nicht ausreichend standardisiert.

Nutzt man serverseitiges Rendering, so kann die Karteninformation gecacht werden und nur die Kacheln, dessen Aktualisierung erforderlich können bei Bedarf neu gerendert werden. Dies bedeutet allerdings, dass größere Datenmengen zwischen Server und Client transferiert werden müssen.

¹ Internet Explorer 8

Es zeigt sich allerdings, dass dieser Verfahren funktioniert.

Sollte der HTML 5-Test negativ ausfallen, so bleibt nur noch serverseitiges Rendering übrig.

Spielbarkeit

Die Spielbarkeit zwischen einer flachen Welt und einer Welt mit Höheninformation unterscheidet sich bei der Bedienung der Karte nur unwesentlich.

Der Nutzer ist aus anderen Spielen gewohnt zweidimensional mit Höheninformationen zu denken. Auch ist die visuelle Darstellung leicht verständlich, da der Nutzer nur in horizontale Richtungen zu scrollen braucht. Sollte ein Berg allerdings ein Objekt verdecken, so muss er die Rotation der Karte um 90° erlernen. Dies ist zumutbar.

Mit Hilfe von Schattierungen (siehe oben) kann die dreidimensionale Struktur der Karte dargestellt werden.

Auch muss der Nutzer verstehen, dass ein Erklimmen von Bergen nicht immer sinnvoll ist und man durch die Täler einen Umweg gehen muss. Dazu ist eine Bereitstellung von Wegfindungen notwendig, da man dem Nutzer nicht zumuten kann jeden Weg einzeln zu definieren.

Möglichkeiten für den Nutzer

Der Nutzer hat neben der Modifikation des Feldtypes auch die Möglichkeit Felder höher oder tiefer zu legen. Dies ist eine kreative Möglichkeit, die auch strategisch genutzt werden kann.

Der Nutzer ist in der Lage Punkte zu kontrollieren, die auf Grund der Höhe und des Feldtypes weitere strategische Vor- und Nachteile bietet.

Weiterhin erscheint das Spiel vollständig transparent, da es den gewohnten Abläufen genügt.

Umsetzbarkeit

Die Umsetzbarkeit dieses Kartentypes erfordert zwingend die Implementierung von Wegfindungs-Algorithmien. Diese kann auf Serverseite implementiert werden, wenn gewährleistet ist, dass die Reaktionszeit < 1 Sekunde ist.

Die Karte kann ähnlich der flachen Karte in einer zweidimensionalen Struktur gehalten werden. Neben des Feldtypes muss allerdings auch die Höheninformation gespeichert werden (2 Bytes [0 - 65 km +. 1m]).

Bei 2-3 GB Speicher können daher Karten mit etwa 25.000 Feldern im Quadrat zur Verfügung gestellt werden. Dies beträgt 250 km bei 10 m Auflösung.

Die grafische Darstellung erfordert wesentlich mehr Aufmerksamkeit und Arbeitsleistung. Allerdings ist kein vollständiger 3d-Renderer erforderlich.

Volumenwelt (3D)

In der Volumenwelt wird ein dreidimensionaler Körper vollständig mit Informationen über die Welt gefüllt. Dies kann in diskreten Element (Voxeln) erfolgen.

Jeder Voxel besitzt eine dreidimensionale Koordinate und besitzt einen Feldtyp.

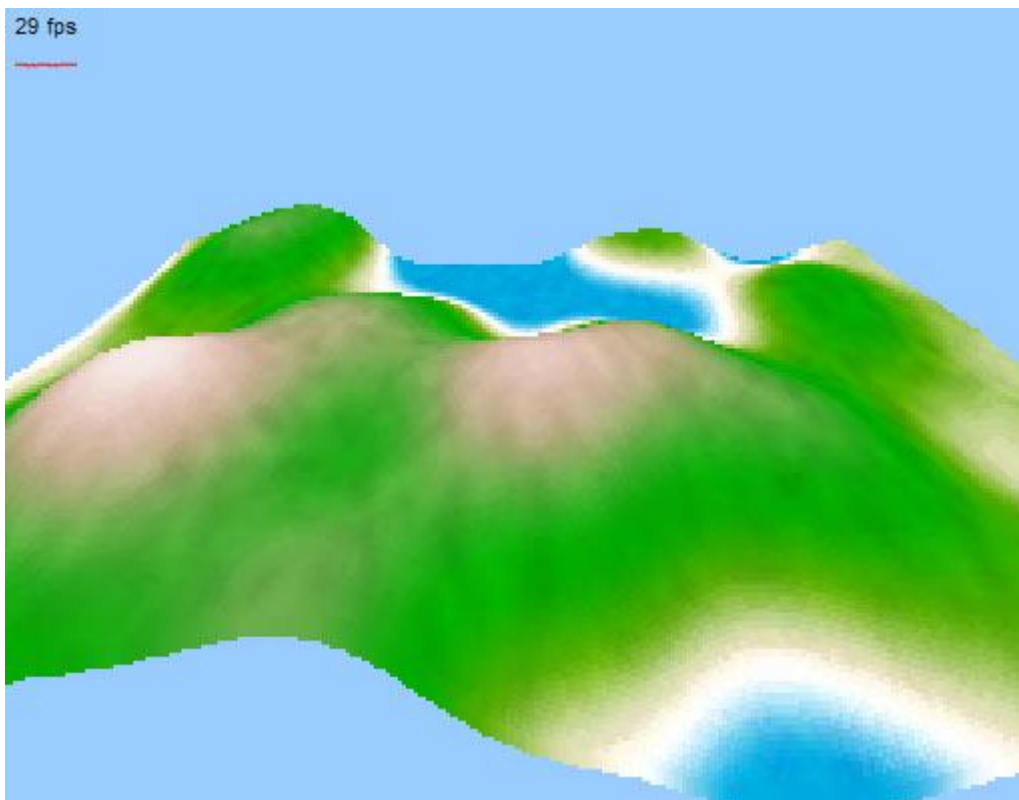
Voxel können auf verschiedene Arten definiert werden:

1. Jede Voxel enthält eine eigenständige Datenstruktur über Feldtyp und weiteren Informationen.
2. Jeder Voxel enthält eine Referenz auf eine Datenstruktur. So können Voxel zusammengefasst werden.
3. Es gibt eine Liste von Datenstrukturen, dessen Zusammenfassung der Welt entspricht. Nichtdefinierte Voxel können als Luft (Standardwert) angesehen werden oder mit einer weiteren Definition standardisiert werden. So kann alles über 0 m als Luft angesehen werden, alles unter 0 m als Wasser.

Dieses Datenmodell wird in Strategiespielen bisher kaum oder nicht eingesetzt.

Objekte besitzen dann ebenfalls eine gewisse Voxelgröße und belegen mehrere Elemente.

Beispielsweise können folgende Bilder gezeigt werden:



Und natürlich dem bekanntesten Vertreter:



Polygonmodelle?

Eine vollständige Darstellung über Polygone überfordert die mögliche Implementierungskomplexität und kann nicht im Rahmen von Depon.Net umgesetzt werden. Hierzu ist auch ein Levelgenerator notwendig, der auch dreidimensionale Polygone erzeugen kann. Dies ist nicht umsetzbar.

Grafische Darstellung

Möchte man eine solche Voxellandschaft zur Verfügung stellen, so sind mehrere Einschränkungen zu beachten:

Die Bedienung auf Touch-Screen-Geräten ist bei einer vollständig freien Ansicht stark eingeschränkt. Der Nutzer ist es nicht gewohnt dreidimensionale Darstellungen für komplexe, räumlich getrennten Abhängigkeiten zu nutzen, wie es für Strategiespiele notwendig ist.

Es ist zu überlegen, ob man die Voxel-Darstellung so implementiert, dass sie nur in einer isometrieähnlichen Darstellung gezeigt wird. Allerdings führt hier der Vorteil einer dreidimensionalen Datenstruktur, dass Kartenbestandteile verdeckt werden.

Fixiert man die möglichen Kameraeinstellungen, so kann die Karte sowohl auf dem Server, als auch auf dem Client gerendert werden. Dazu ist aber ein Auswahl von Karteneinstellungen zu wählen, die den Nachteil der möglichen Verdeckung vermeidet. Dies erscheint schwierig.

Es ist daher also davon auszugehen, dass auf dem Client gerendert werden muss.

Spielbarkeit

Die Spielbarkeit einer vollständig dreidimensionalen Landschaft ist sehr stark eingeschränkt.

Es ist einem Spieler in einem Strategiespiel nicht zuzumuten, dass er in der Ego-Perspektive oder andauernd bei drehender Karte Positionen und Voxel auswählt.

Ist die Karte fixiert, so kann er keine Voxel unterhalb der Oberfläche auswählen.

Einschränkung der Regeln

Wenn allerdings definiert wird, dass sich Objekte nur oberhalb des höchsten, gefüllten Voxels befinden dürfen und unterirdische Elemente zur Speicherung bestimmter Rohstoffe (Erdöl-Felder oder ähnliches) genutzt werden, so ist die Karte ähnlich bedienbar wie die 2.5D-Karte.

Weiterhin bietet diese Karte den Vorteil, dass Tunnel und Brücken einfach zu implementieren sind.

Möglichkeiten für den Nutzer

Ist definiert, dass sich der Spieler unter die Erde graben darf, so ist er völlig frei in seiner Gestaltung.

Ist der Spieler allerdings eingeschränkt, so kann er die Karte ähnlich der 2.5D-Karte mit Höheninformationen nutzen. Hierzu sind ebenfalls Assistenzsysteme, wie Wegfindung erforderlich.

Ist ihm erlaubt Brücken oder Tunnel zu bauen, so kann er strategische Objekte selbst erschaffen und so einen großen, überraschenden Einfluss auf das Spielgeschehen ausüben.

Weiterhin muss bei der Umsetzung darauf geachtet werden, dass kein Spieler über Tunnel oder Höhlengänge in der Lage ist unfaire Aktionen inmitten der Verteidigungslinien durchzusetzen. Depon.Net 2.0 ist ein defensives Spiel², bei der ein Spieler sich auf seine Verteidigung verlassen kann.

Er kann kreative Elemente nutzen und auch die Objekte selbst können aus mehreren Voxeln bestehen.

Die Definition und Implementierung eines Kartensteuerungsmodell ist allerdings eine Herausforderung mit großem Risiko.

Umsetzung

Die Umsetzung ist das größte Manko bei diesem Kartentyp.

² auch wenn es dafür bisher keine Anforderung gibt

Servergenerierte Karten fallen wegen der obig geforderten, freien Kartenbewegung aus.

Die Verwendung von SVG oder Canvas muss geprüft werden. Technologiestudien³ zeigen zwar, dass es man angenehme Performance erreichen könnte, es muss allerdings geprüft werden, ob ich selbst dazu in der Lage bin. Insbesondere texturierte Voxel können hier interessant werden.

Wird die Karte allerdings nur statisch generiert und dann nur auf einer Ebene bewegt, so fällt ein langsames Rendering nicht sonderlich auf, da die Karte auf dem Browser vorgeneriert werden kann und nur noch verschoben werden muss.

WebGL ist nicht ausreichend verbreitet. Hier gibt es allerdings schon eine Implementierung, wie ich sie gerne hätte.⁴

Auf Serverseite muss eine dritte Dimension ebenfalls gespeichert werden.

Bei 100 Höheneinstufungen und 2-3 GB Speicher kann nur noch eine Karte von etwa 1.700 in Quadrat erzeugt werden. Ist jeder Voxel etwa 10 m groß, so können nur noch 17 km gezeigt werden.

Hier ist ebenfalls die Implementierung einer speichersparenden Implementierung erforderlich. Beispielsweise kann ein Voxel im Körper den Typ aller darunterliegenden Voxel definieren. Befindet sich dort wieder ein Voxel anderen Types, so beginnt ab hier eine neue Bodenschicht. So können bei durchschnittlich 5 Bodenschichten und 2-3 GB Speicher eine Karte von etwa 15.000 Pixel gespeichert werden. Dies entspricht 150 km.

Migrationsszenario

Es ist ebenfalls zu überlegen, ob man auf Serverseite eine reine Voxelkarte speichert, während man auf Clientseite nur eine Karte mit Höheninformationen zeigt. So ist eine isometrische Darstellung notwendig, während man sich die Möglichkeit offen hält weitere Strukturen Schritt für Schritt einzuführen. Auch ist der Implementierungsaufwand überschaubar.

³ http://mrdoob.github.com/three.js/examples/canvas_geometry_terrain.html

⁴ http://mrdoob.github.com/three.js/examples/webgl_geometry_minecraft_ao.html