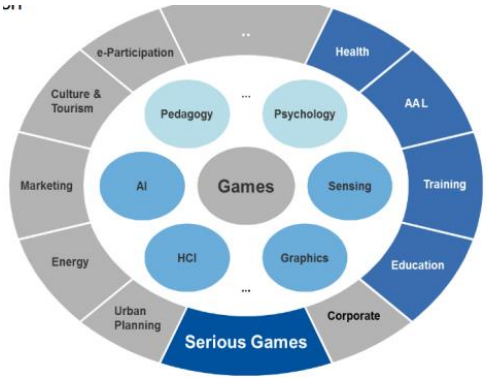
SG Klausurinhalte

Contents

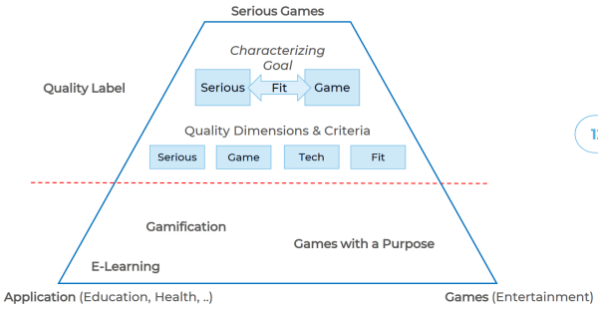
Introduction

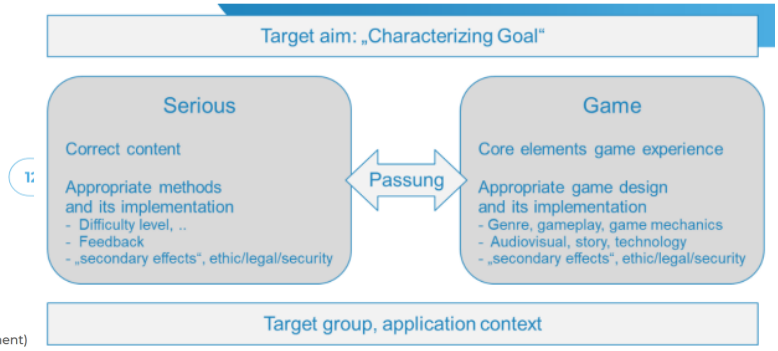
Definition Serious Games: Spaß + charakterisierendes Ziel



Lernspiel: Ein Lernspiel ist ein (digitales) Spiel, das entwickelt wurde, um zu unterhalten und dessen chracterizing goal darüber hinaus „Lernen“ ist. Lernspiele möchten Wissen und Fähigkeiten vermitteln und stellen einen Teilbereich der Serious Games dar.

Gütekriterien

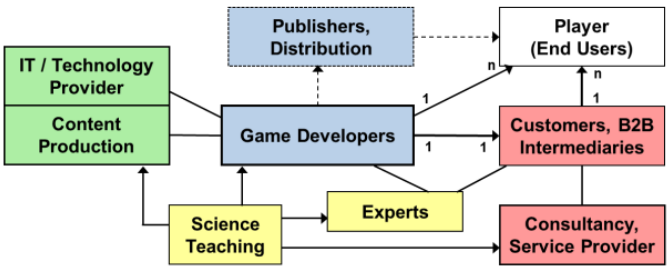




Serious Games vs Gamification

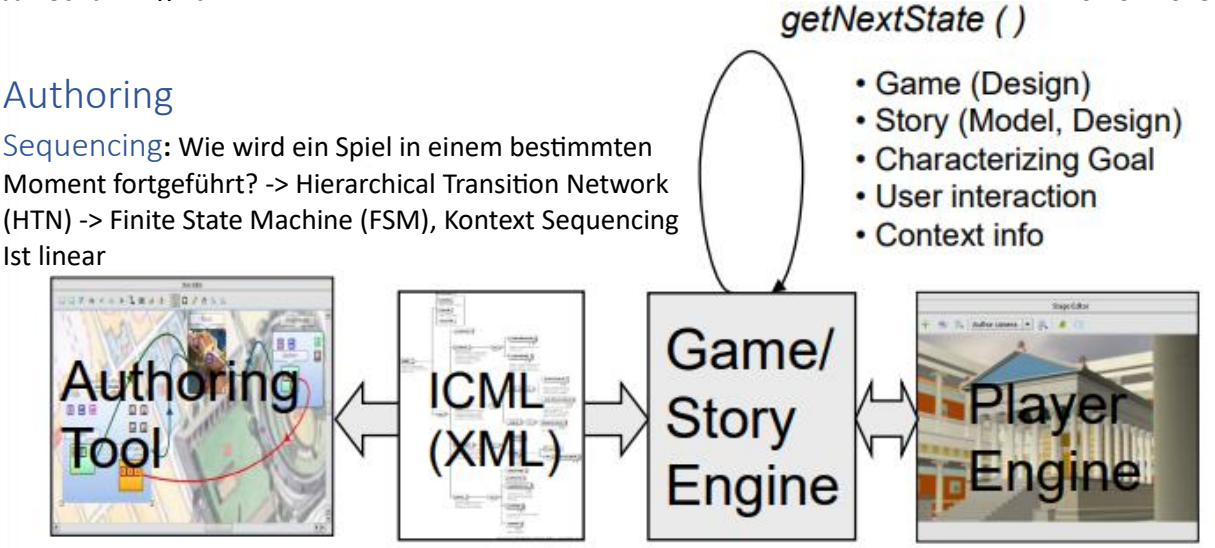
Gameification ist das hinzufügen von Spiel-Elementn in einen nicht-Spielbereich, Gameification an sich ist kein Spiel.
Elemente für Gamification können sein:
Punkte (Fortschrittvisualisierung, Feedback)
Badges (Pointification) (Zielsetzen, Anweisung, Reputation, Gruppenidentifikation)
Bestenlisten (Pointification) (Vergleich, Ranking)

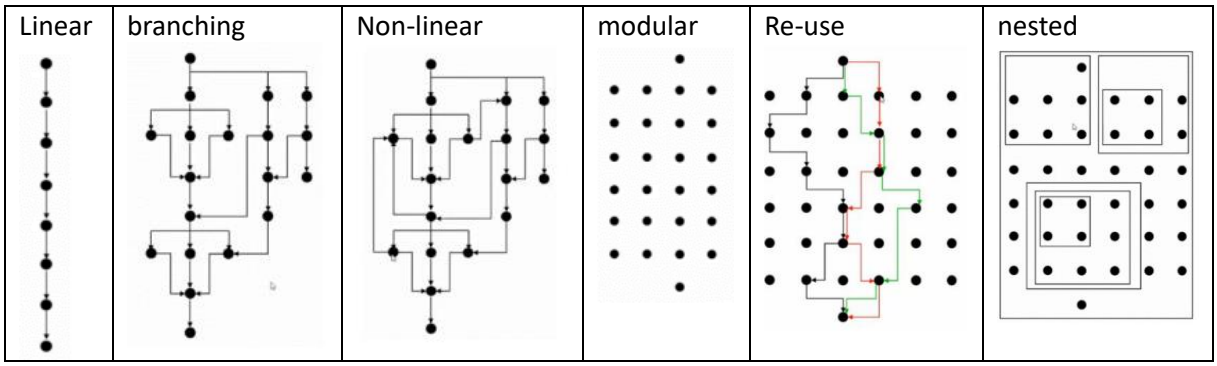
Geschäftsmodell Serious Games



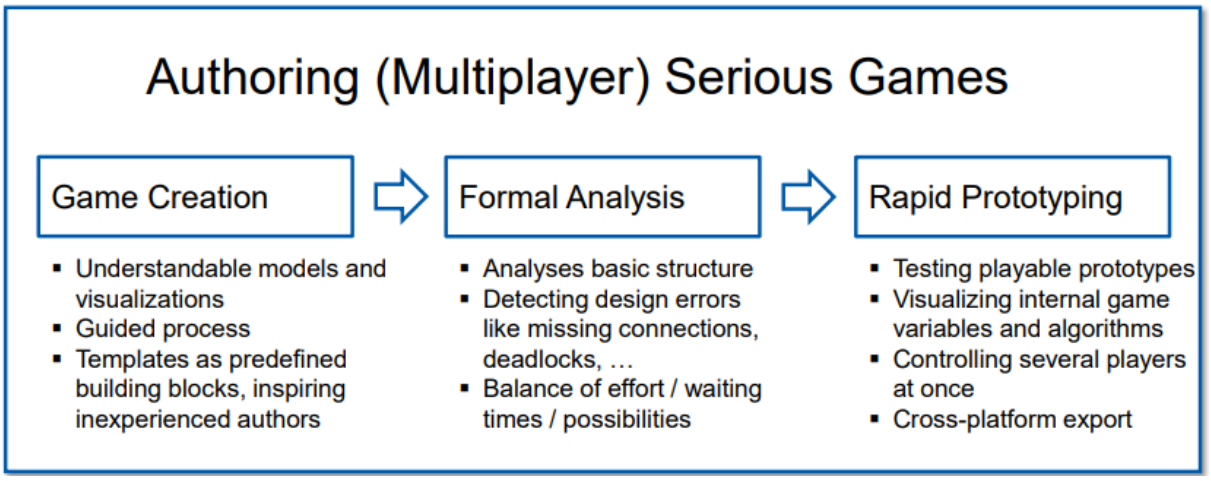
Authoring

Sequencing: Wie wird ein Spiel in einem bestimmten Moment fortgeführt? -> Hierarchical Transition Network (HTN) -> Finite State Machine (FSM), Kontext Sequencing
Ist linear





Authoring Process



NGLOBs

NGLOBs (Narrative Game-Based Learning Objects) sind eine Methode für Sequencing. Eine Szene eines Spiels bekommt 3 Dimensionen von Annotationen mit numerischer Annotation 0 bis 1. Aus den Annotationen kann man die Geeignetheit der Szene zum aktuellen Zeitpunkt berechnen.

1. **Narrative**: Storymodel und Eignung für einen Punkt in der Geschichte
    a. SM_HJ_1, 0.1 Storymodel nach Hero's Journey Abschnitt eins hat eine Eignung von 0.1
2. **Game-based**: Playermodel Eignung, Bartle: Killer, Achiever, Socializer, Explorer
    a. PM_BA_K, 0.5 Playermodel nach Bartle, Eignung Killer 0.5
3. **Learning**: Welche Fähigkeit ein Spieler als Voraussetzung benötigt und welche er lernt. Als Grundlage dient ein Knwoledge Space. Eine Szene kann „associated Skills" (lerndene Skills) oder „Prerequisite Skill" (Voraussetzende Skills) haben. Kompetenzbaum
    a. {A101, A102}, {P210, P217} Verweis auf Knoten 101 im Knowledge Space (associated Skill)

$$\underbrace{\left\langle \begin{matrix}(SM\_HJ\_1, 0.1),\\ (SM\_HJ\_4, 0.2),\\ (SM\_HJ\_8, 0.1)\end{matrix} \right\rangle}_{\text{Story Model}}, \underbrace{\left\langle \begin{matrix}(PM\_BA\_K, 0.2),\\ (PM\_BA\_S, 0.4),\\ (PM\_BA\_E, 0.9)\end{matrix} \right\rangle}_{\text{Player Model}}, \underbrace{\left(\begin{matrix}\langle A101, A102\rangle\\ \langle P210, P217\rangle\end{matrix} \right)}_{\text{Learning Context}}$$

Authoring Konzepte

Mit Middleware verbunden	• StoryTec Autoren-umgebung • Autoren können mit geringem Know-How Inhalte erstellen • Game Engines zu schwer
Novice Users	• Abstraktion • Vordefinierte Strukturen, Programmierung nach Beispiel • Natural Language Programming
Process Support	• Metadaten Vordefinierte Workflows
Iterative Herangehensweise	• WYSIWYG • Schnelle Prototypen
Complexitäts trade-off	• Generisch: flexibel aber komplex • Spezifisch: Einfach zu nutzen, weniger Features

Elemente von Authoring Konzepten (Storytec)

- **Stage Editor** (WYSIWYG Editor, Schnelles bearbeiten, Verstecken von Spielmechaniken, Template visualisierung
- **Objects Browser**
- **Story Editor** (Visualisierung der Struktur und übersicht über das Projekt, Hierarchie und Transitionen von Szenen)
- **Property Editor**
- **ActionSet Editor** (Actions programmieren, basierend auf Objekten – Branching durch Conditions)
- **Knowledge Space Editor** (Erstellen eines Skill-Trees / Kompetenz-Space)

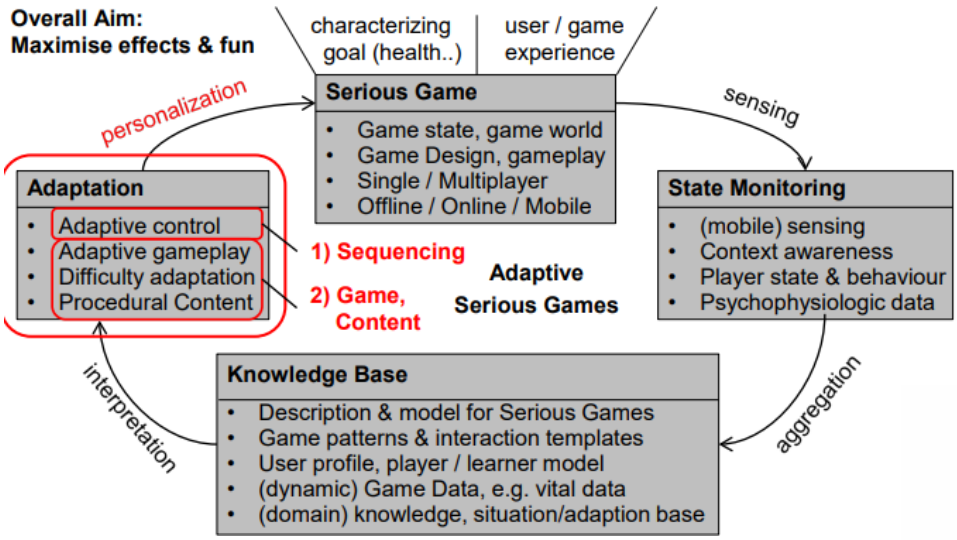
Unterschied Authoring / Content Creation

Digital Content Creation (DCC) Tool	**Authoring Tool**
• Erstellt, um die Erstellung von Inhalten zu ermöglichen • Endresultat ist nur ein Teil des gesamten Produkt • produziert individuelle Assets	• Erstellt, um Authoring zu ermöglichen • Endresultat ist das finale Produkt oder ein großteil davon • Zusammenfassen von durch das DCC erstellten Assets

Formale Analyse

Model Checking: automatische die Korrektheit eines Modells im Vergleich mit den Spezifikationen prüfen, meist mit Hilfe von Petrinetzen, dadruch können folgende Fehler gefunden werden: Unused variables, Islands, Dead Ends, Unsatisfiable Conditions

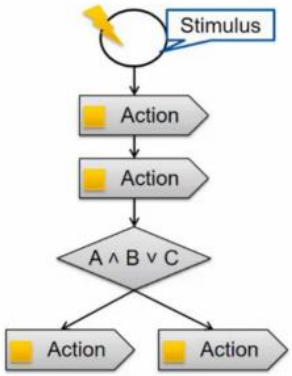
Personalisierung und Adaption:



Interaktivität

Interaktivtät ist wie folgt definiert:

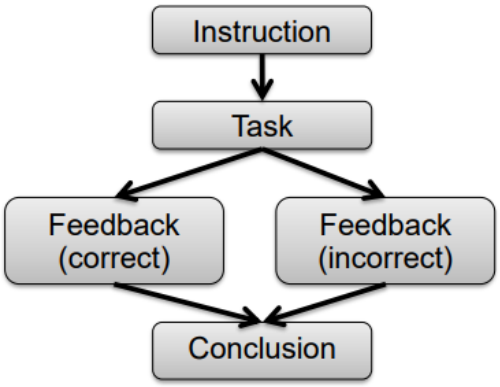
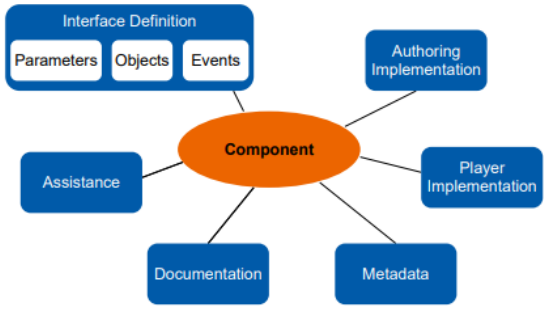
- Stimuli: High Level User Events, verwendet um Reaktionen zu definieren
- (Re-)Aktionene: Definiert für Objekttypen (etwas sichtbar machen, sound abspielen,...), High Level Commands



- Action Sets: Kombination aus Aktionen, linearer Ablauf der Aktionen, verwendet boolean Conditions

Authoring – Templates

Jedes Genre hat sein Set an Game Design Patterns, wiederholenden Elementen etc.

Structoral Templates	Interaction Templates
• Vordefiniertes Set an Szene, Objekten und Übergängen • Wiederverwenden von gängigen Strukturen • Guideline für Autoren 	• Bietet komplexes Gameplay oder Interaktion • Verschachteln der relevanten Parameter, Objekte und Events • Präsentation von Inhalt separiert • Als Komponente realisiert 

Game Design

Ein Spiel besteht aus den 3 Bereichen Genre, Theme und Setting.

Theme: Welches Gefühl das Spiel vermitteln soll: Angst, Freiheit, Macht, Kontrolleurm etc.

Setting: Geografisch oder Zeitlich: Post-Apokalyptisch, Western, Sci-Fi, Modern, Mittelalter etc.

Genre

Das Genre wird durch Mechaniken und Spielerinteraktion definiert

Spielanalyse: Analyze – Learn – Recreate: Genres verstehen, Themen verstehen, Erfolg analysieren

Genre erstellen:

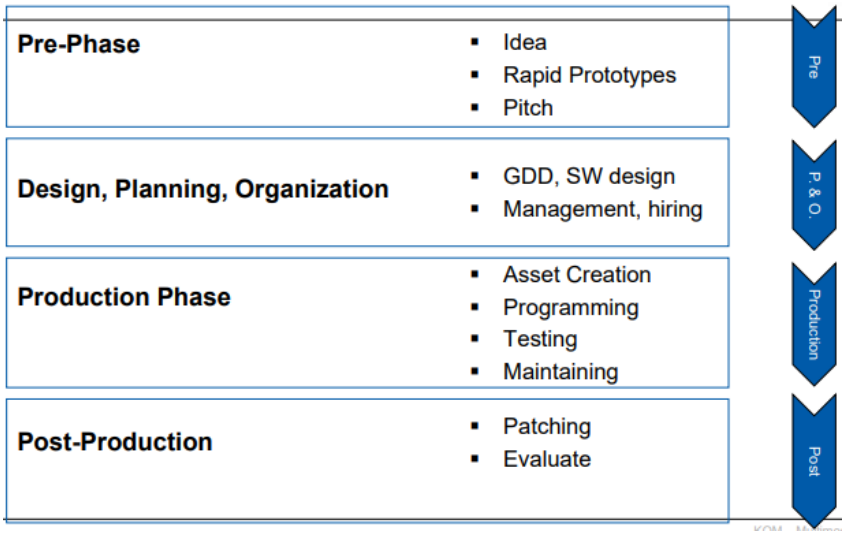
- Soziale Konvention: Gesellschaft entscheidet, welchem Genre das Spiel angehört, Ggf. neues Genre erfinden
- A-priori Methode: Leute arbeiten zusammen und erstellen Checkliste
- Idealist Method: EIN Spiel definiert ein ganzes Genre (bsp: Roguelike, Soulslike, JumpNRun)
- Empiric Method: Alle Spiele nehmen und nach Ähnlichkeiten clustern

Genrearten:

Action	Physiche Herausforderungen (Hand-Auge-Koordination, Reflexe, Spieler hat Kontrolle) Subgenre: Platformer, Shooter, Fighter
Adventure	Gameplay ohne Reflex-Herausforderungen, Rätsel lösen, Interaktion mit anderen Charakteren ohne Kampf Subgenres: Text adventures, Visual Novels
Roleplay	Ursprung in Tabletops und Pen'n'Paper: EXP->Level-Up->Skill mit Optionen für Chartakterentwicklung Subgenres: Action RPG, MMORPG, Roguelike

Simulation	Realität simulieren, Simulation managen Subgenres: Konstruktion, Leben, Fahrzeuge
Strategie	Planung, Allgegenwärtiger Spieler, Rundenbasiert vs Echtzeit Subgenres: 4X, Autochess, MOBA, RTS
Cross-Genre	Spiele können Genres kombinieren um so viele Spieler wie möglich zu befriedigen, Aspekte mehrerer Genres um mehr Spieler anzulocken. Die Funktionen der verschiedene Genres müssen miteinander verschmelzen und synergieren.

Game Design Prozess



Rollen

Production	**Game Design**	**Asset Creation**	**Game Programming**
• Producer (director), Publisher • Individuelle Leiter für jede Disziplin	• Designer • Writer	• Artist • Sound Engineer • Level Designer	• Engine Programmer • Tool Programmer • Gameplay Programmer • UX Designer

Inde vs AAA

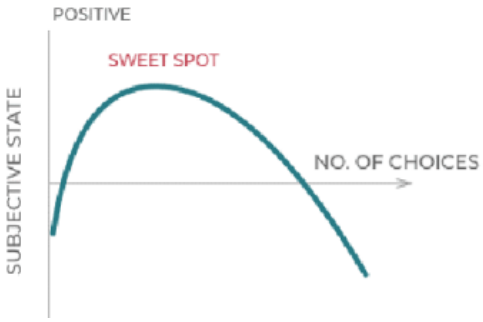
Indie	**AAA**
• Kleines Team • Niedriges Budget • Kurze Entwicklungszeit ~ 1 Jahr • Frei in Design Entscheidungen	• Große und mehrere Teams • Großes Budget • Lange Entwicklungszeit ~ 3 Jahre • Design durch den Markt bestimmt
Scope • Simple, kostenlose Tools verfügbar: Game Maker, Unity, Unreal • Aufwandsminimierung: Variation einfacher Spiele, Story, inoovatives Gameplay, kein 3D, aufwendige Grafik • Finanzierung: eigenes Budget, Kickstarter • Ermöglicht durch digitale Verteilung: Steam, Desura, Xbox LIVE, Appstore • Niedriges Risik = mehr Innovation? Muss nicht dem Massenmarkt gefallen, wichtig für serious Games	**Scope** • Längere Zeitspanne • Viele Menschen im Hintergrund • Komplexe interdependencies, externe Faktoren • Multimillionen-Budget: Salaries, Lizensen, Hardware, Mietkosten • Hohes Risiko, formalisierter Prozess

Entwicklungsteams, Outsourcing: Nicht jede Rolle wird gleich viel benötigt, machmal auch spezielles Know-How, Outsourcing von Übersetzung, Sounds, Cutscenes etc.

Level of Design

Macro Level	Micro Level
• „High“ Level • Story Modus • Generelle Strategie verfügbar • Outline • Spielstil, Map, Mode, Level aussuchen	• „Low“ Level • Single Choice • Mico managment Tactic • Detail • Skillpoints, Dialogue options, Food etc

Paradox of Choice & Meaningful Choices



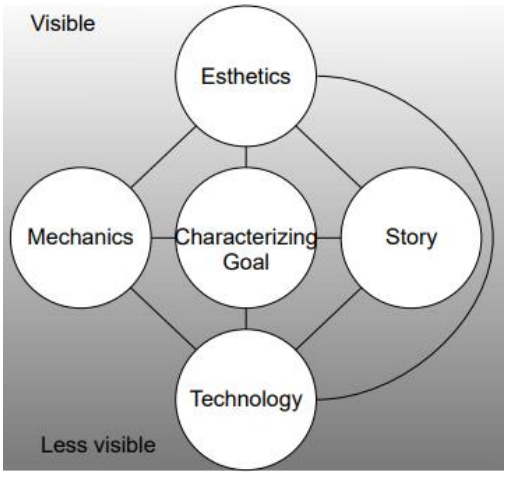
Zu viele Möglichkeiten können den Spieler überfordern, weniger Optionen bieten einen größeren Impact pro Option
Entegegenwirken: Spieler im Entscheidungsprozess unterstützen und verschiedene Möglichkeiten bieten
Optionswahl:

1. Eigene Ziele definieren
2. Die Wichtigkeit jedes Ziels evaluieren
3. Optionen auflisten
4. Jede Option zum Erreichen des Ziel evaluieren
5. Beste Option aussuchen
6. Konsequenzen identifizieren, um dabei für spätere Entscheidungen zu lernen

Game Design Document (GDD)

Konzept Dokument	Design Dokument	Technisches Dokument
• Breite Definition, charakterisierendes Ziel • Pitch • Genre • Setting • Theme • Basic Gameplay • Zielgruppe	• Detaillierte Beschreibung aller Features (Player actions, inputdevice, Story) • Rechtfertigung der Designideen • Jeder Developer sollte das Dokument einsehen können	• Alles mit Hardware und Software • Zielgerät • Restriktionen (Low End, High End, Formate) • Verwendete Tools • Convetionen (Naming, Resolutions, Coding)

Game Tetrad (Pentrad)



Spiele bestehen aus 4 Elementen: Mechaniken (s. Spielmechaniken), Esthetics, Story (s. Storytelling) und Technologie (s. Flow, Multiplayer, Level of Design), Serious Games erweitern diese mit dem Characterizing Goal

Spielmechaniken

Folgende Mechaniken gibt es: Gamespaces, Aktionen, Regeln, Fähigkeiten, Wahrscheinlichkeiten, ...

Gamespaces

Gamespaces können Diskret (Bewegungen innerhalb eines Schachfeldes sind kein neuer Gamestate) oder Kontinuierlich (Jeder Positionswechsel ist ein neuer Gamestate) sein. Gamespaces können aber auch Multidimensional sein oder limitierend, sogar verschachtelt. Nested Game Spaces: Manchmal benötigt man nicht die gesamte Außenwelt, daher den Space in kleinere Subspaces unterteilen und mit der Menschlichen Denkstruktur anpassen (In-House, Outside, On-Road, ...) dies bringt auch einen Performance boost mit sich, da nur der relevante Space simuliert werden muss.

Gamespace Abstraktion

Menschen haben Schwierigkeiten Kontinuierliche Spaces zu kommunizieren, daher werden diese in Logik und Code so abstrahiert: If (x >= 0.5 && x < 0.8). Strategien, AI und allgemeines Verständnis ist in diskreten Spaces einfacher zu verstehen, zB A* Pathfinding. Daher sollten kontinuierliche Spaces in ein diskreten Space abstrahiert werden, auch wenn es nur in der Design Phase ist.

Aktionen

Operative Aktionen	**Implizite, resultierende Aktionen**
• Springen, Bewegen, Essen • Einzelne Aktionen	• Veränderung des Gamespace/ Objekte • Oft Teil von der Strategie • Für beabsichtigte Aktionen (Bewegen von Objekten um den Weg frei zu machen, Angreifen um Schaden hinzuzufügen) • 2. Dimension der operativen Aktion: eine operative Aktion sollten immer mehrere implizte Aktionen zugewiesen werden.

Regeln

Operationale	**Fundamentale**	**Behavioral**	**Geschriebene**
• Wie man das Spiel spielt • Objektive Beschreibung	• Mathematische Repräsentation der umgangsprachlichen Operationalen Regeln • Veringern eines Wertes erhöht einen anderen	• Mehr Ethiken: In den Schachregeln steht nicht dass man den Gegner nicht schlagen darf	• Anleitung, Tutorial

Fähigkeiten

Virtuel	**Real**	**Physisch**	**Mental**	**Sozial**
• Skillpoints • Klassen • Tech-Skill	• Physisch • Mental • Sozial	• Stärke • Ausdauer • Reflex	• Gedächtnis • Mustererkennung	• Kommunikation • „lesen“ des Gegners

Wahrscheinlichkeiten

Wahrscheinlichkeiten werden zum Balancieren von Spielen verwendet. Wenn man zwei Wahrscheinlichkeiten mit einem UND verbindet, muss man die Wahrscheinlichkeiten multimpliziernen, wenn man sie mit einem ODER verbindet muss man sie addieren. Der **Erwartungswert** beschreibt, welches Ereignis am häufigsten auftritt.

Erwartungswert (Predictand)

$P(X = x_i)$ ist die Wahrscheinlichkeit, dass Ereignis x_i eintritt.

$$E(X) = x_1 \cdot P(X = x_1) + x_2 \cdot P(X = x_2) + \cdots + x_n \cdot P(X = X_n) \Leftrightarrow E(X) = \sum_i (x_i \cdot P(X = x_i))$$

Wird für eine grobe Balancing Annahme verwendet, z.B. für (Micro)Transactions, Lootboxes, Generelle Berechnungen für Glücksbasierte Mechaniken.

Varianz

Die Varianz gibt an, wie sich deine Beobachtungswerte um den Mittelwert aller Beobachtungen verteilen.

$$Var(X) = E(X^2) - E(X)^2 \Leftrightarrow Var(X) = E([X - E(X)]^2)$$

Standardabweichung: $S(X) = \sqrt{Var(X)}$

Je geringer die Standardabweichung ist, desto zuverlässiger ist die Mechanik.

Balancing Ansatz

Wie balanciert man z.B. zwei Tränke die gleich stark sein sollen. Wir verwenden die Notation 1d6 um zu beschreiben, dass wir einen sechsseitigen Würfel verwenden.

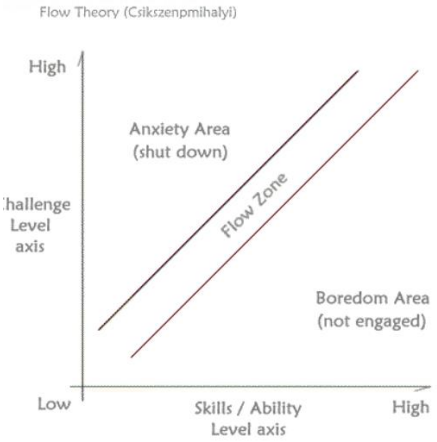
$E(B) = E(1d6 + 3) = 6.5$
$E(A) = E(1d6) + 0.1 * E(1d6 + 3) = 4.15$
Now we want $E(A)$ to be 6.5

$$6.5 = E(1d6) + E(1d6 + 3)x$$
$$E(1d6 + 3)x = 6.5 - E(1d6)$$
$$x = \frac{6.5 - E(1d6)}{E(1d6 + 3)}$$
$$x = \frac{6.5 - 3.5}{6.5} = \sim 0.46$$

Therefore: If extra roll has a probability of 46% both potions are equally powerful
BUT: $S(B) = 1{,}7078$
$S(A) = 3.84 > S(B)$

Murphys Law

Murphys Law besagt, dass alles schief gehen wird, was schiefgehen kann (Unabhängige Wahrscheinlichkeiten). Wenn man z.B. nie den passenden Münzwurf bekommt, kann man nach einer bestmmten Zahl Fehlversuchen die Wahrscheinlichkeit für einen richtigen Versuch erhöhen aber Achtung, dieses System kann ausgenutzt werden.



Flow

Flow ist eine Zone, in der der Spieler voll in der Immersion ist und einen genauen Fokus auf das Spiel hat. Er wird mit Spaß im Prozess seiner Aktivität bereichert. Es ist die komplette Absorption des aktuellen Handlungsprozess.

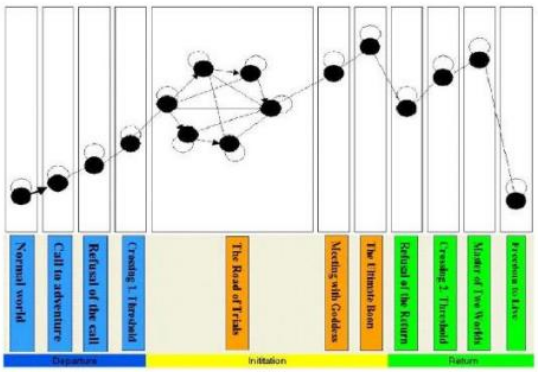
Schwierigkeit	**Feedback**	**Clear Goals**	**No Distraction**
• Verschiedene Spieler bringen verschiedene Skills und Motivation • Zielgruppe erkennen • Ursprung der Schwierigkeits-einstellung • Kategorisieren in den Einstellungen ermöglichen • Adaption: Spiel passt sich automatisch an, schwer mathematisch zu evalualieren • Falsche Schwierigkeit führt zu Angst oder Langeweile	• Motiviert den Spieler • Arten: Visuell, Auditiv, Haptisch, Item / Currency • Feedbackarten sind multidimensional • Visuelles Interface: sofortiges Feedback in der Nähe des Fokuspunktes • Spiel ohne Feedback wird langeweilig	• Drei Kategoiren: Explizit, Implizit, Playermade • Explizit: Main Quest / Side Quest • Implizit: Keine explizite Erwähnung • Playermade (am meisten effizent aber schwer zu designen): Unterstützen aber geben nichts vor • Unklare Ziele führen zu „Doodeling“ und frustration	• Keine unnötige Ablenkung erstellen, die den Spieler vom Ziel ablenkt • Spieler nicht mit Informationen, Optionen überwältigen

Storytelling

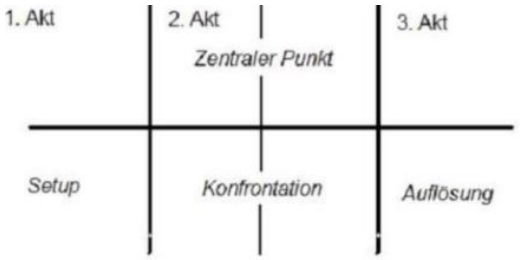
Aristoteles: Für lineare Geschichten, bietet die Grundlage für alle, besteht aus drei Akten: Exposition, Steigende Handlung bis zum Höhepunkt, Auflösung

Heros Journey: Adaption und vereinfachung für Filme in 12 Schritten, konzentration auf Elemente für Spannung

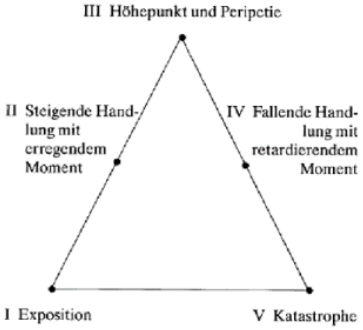
	1.Gewohnte Welt Exposition
	2.Ruf des Abenteuers
I. Akt	3.Weigerung
	4. Mentor
----------	5. erste Schwelle, 1. Wendepunkt
	6. Proben, Verbündete, Feinde
	7. Vordringen zur tiefsten Höhle
II. Akt	8. entscheidende Prüfung
Höhepunkt	9. Belohnung
----------	10.Rückweg, 2. Wendepunkt
III. Akt	11.Auferstehung
	12.Rückkehr mit dem Elixier

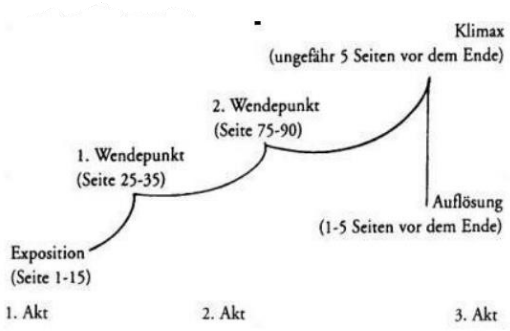


Syd Field: Modell zum erstellen eines Skripts, erweitert Aristoteles Modell mit Punkten aus modernen Drama, Skriptseiten (Filmminuten) als temporale Struktur



Gustav Freytag: Fünf-Akt-Pyramidenstruktur

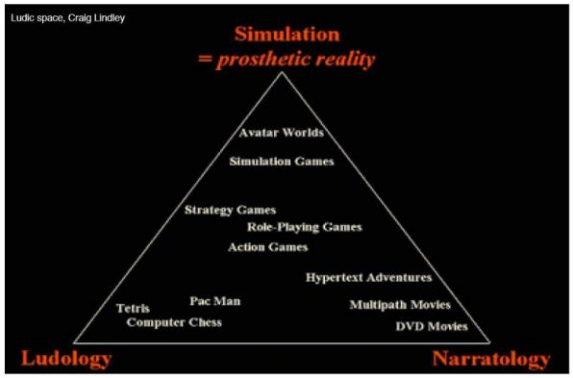




Linda Seger: Change Points steuern die Geschichte in eine Richtung mit steigender Spannung und Gefahren hin zu einem Höhepunkt.

Propp: Kategorisierung von Story Units, Einführung von „Dramatis Personae“, Struktur für Märchen folgt morhpohologischen Funktionen, diese Funktionen sind immer in der selben Reihenfolge, Alle Charaktere sind Funktionen, 31 erzählende Elemente mit 7 „Zyklen/Unit“

Narrative paradox: Analyse von existierenden Systemen und Herangehensweisen, Fokus auf Balance zwischen „player control (Interactivity)“ und „dramatic/ author control (narrative)“



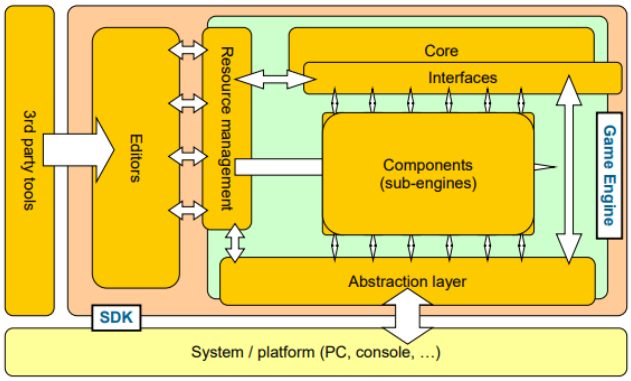
Storytelling Systeme:

Plot-based:	Emergent character-based	Guided character-based
• Narrative Struktur	• Charaktere mit Verhalten	• Kombinierter Ansatz

• Narrative Control > User Interaction • Ziel: Spannung, Dramatourgie passend zur Storystruktur • Authoring: Unterteilt in Story Elemente, verbindung durch Events • Vordefinierte Pfade • Interaktion nur auf Erzählerlevel	• Story kommt wöhrend des Spielens zu stande • Story basiert auf Charakteristiken und Aktionen des Charakters • Keine storystruktur • User Interation > Narrative Control	• Überwindet die Limitaitonen beider Ansätze • Narrative Control + User interaction

Unity

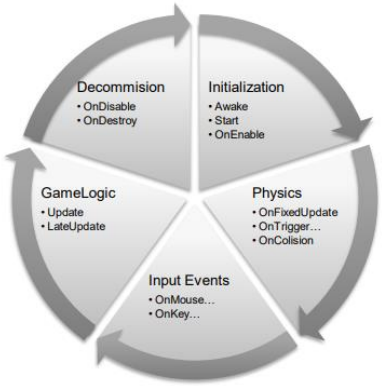
Game-Engine Architektur



Core layer: Spiellogik, Controller Komponent
Komponenten layer: Kapselt bestimmte Engine Aufgaben
Abstraktions layer: Versteckt das tatsächliche System
Ressourcen Management: Persistente Daten
Domain-spezifische Game Engines: RPGMaker, Game Maker, Cryengine. Unterschied ist, dass Domain-spezifische Engines Tools bietet für das jeweilige Genre oder Domäne.

Game-Loop und Physics-Loop

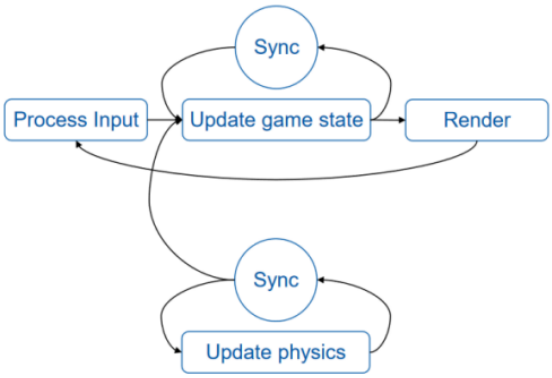
Rendering dauert länger als Update des Game states, nur die Game States müssen synchron bleiben. Lösung ist eine synchronisierter Loop für den Gamestate, für das rendern gibt es einen separaten unsychronisierten Loop. Auch die Physic Loop ist separat, da diese einen Konstanten DeltaTime benötigt und Performance optimiert, hingegen reicht für den Game Loop ein variabler DeltaTime.



Unity Game Loop (simplified)

Funktionen:

- Awake(): Vor dem ersten Update
- Start(): Am Anfang des nächsten frames, ein einziges mal
- Update(): Einmal pro frame
- LateUpdate(): Einmal pro Frame, nach Update()
- OnEnable(): Direkt nach Awake(), immer wenn ein Objekt aktiviert wird
- OnDisable(): Nach rendering cor dem nächsten Frame, immer wenn ein Objekt inaktiv wird
- FixedUpdate(): Vor jedem Update und Kollisionsdetektion, einmal pro PhysicsUpdate



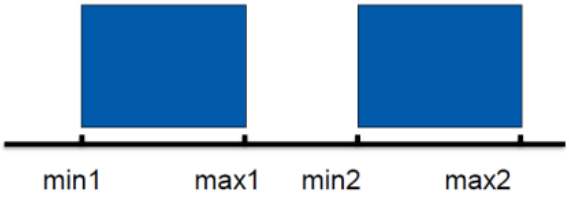
Kollision

Da der Renderstate nach dem Game Loop folgt und in dem Frame kein Physics Update stattfindet, wird das Objekt in dem anderen Objekt gerendert. Wenn dann ein Physics Update folgt, wird die Kollision behoben und das Objekt auserhalb des anderen Objekts gerendert.

Sphere-Sphere	**Plane-Sphere**	**Sphere-Dreieck**
• Kollision, wenn Distanz zwischen Mittelpunkten	• Fläche in der hessischen Normalform	• Kollision mit jedem Dreieck des Meshs prüfen

kleiner als Summe der Radien		• Teuer • Für Dreiecke -> SAT

Ein **Rigidbody** aktiviert die Physik, der Collider erkennt die Kollision, sendet eine Nachricht an den Rigidbody und dieser sendet eine Nachricht an das GameObject, damit ein Physikupdate durchgeführt wird. Ein **Static Collider** ist ein 3D-Objekt, welches kein Rigidbody hat und einen (non-trigger) Collider.
Trigger: In Unity ist ein **Trigger**-Collider ein Collider, der keine physische Präsenz in der Spielwelt hat, aber dennoch Kollisionen mit anderen Objekten erkennen kann. Ein **Non-Trigger**-Collider ist ein Collider, der eine physische Präsenz in der Spielwelt hat und Kollisionen mit anderen Objekten erkennen kann. En Trigger gibt den zugehörigen Collider, die Collision gibt das Collision-Objekt.
Beim **Raycasting**, wird die Kollisionsdetektion mit einem Strahl gemacht und ohne Updates ausgeführt, die MaxDistance ist einstellbar. Da die Physics Loop mit einem festen timestep läuft, läuft diese meist langsamer als der GameLoop und definiert die FPS von FixedUpdate().
Nachdem eine Kollision erkannt wurde, wird eine neue separierende Velocity und ein **Impuls** der die Geschwindigkeit passend ändert berechnet, die Interpenetration wird gelöst und der Impuls wird angewandt.
SAT (Separating Axis Test):
Es muss Punkte P1 und P2 der objekte 1 und 2 geben, sodass die Richtung von P2-P1 eine separierende Achse bildet. Eine separierende Achse projeziert alle Punkte eines Objekts auf die Achse, man berechnet die minimalen und maximalen Punkte min1, min2, max1 und max2, die Objekte sind separiert, wenn max1 < min2 oder max2 < min1.



Broad Phase	Narrow Phase
• Vereinfachte Kollisionserkennung • So viele Kollisionen wie möglich vermeiden • Narrow Phase sollte nur aufgerufen werden wenn Speratation hier nicht bewiesen werden kann • Reduzieren des Problems: Spatial Data Struktures, Bounding Volumes, Collision Matrix	• Exakte Kollision prüfen • Exakte Tests verwenden (SAT) • Deutlich langsamer als die Broad Phase • Beitet Kollisionsdaten für den Resolver • Allgemein sollten Kollisionsberechnungen vermieden werden.

Multiplayer

Erweitert die Motivation bei Serious Games, hat einen sozialen Faktor und trainiert social skills durch **Kommunikation** (In-Game Zeichen, Chat, Voice Chat, evtl Limitieren), fördert **kollaboratives Lernen**. **Game Mastering** beschreibt das beeinflussen des Spielgeschenes durch eine höhere Instanz. Soziale Aspekte von Multiplayer können aber auch negativ sein, wie **Griefing** (Absichtliches nerven anderer Spieler), Account Sharing und Multi Accounting.

Cheating

Problem: Kunden bezahlen für Spaß, Spiel muss fair sein, daher müssen Cheats / Exploits verhindert werden.
Design Issues: Spielerzahl, Beständigkeit von Spielwelten, Lobby /Matchmaking, Game Speed und Flow, Netzwerk, Hardware, E-Sports
Eine **Lobby** ist meistens eine separierte Instanz zum Spiel, ihre Aufgaben sind Statistiken, Shop, Freundeslisten etc.

Application Layer

Exploits/Bugs	**Information Exposure**	**Aimbots**
• Duping • Geometriebasiert (Glitches)	• Maphacks • Skin cheats • Removals	• Farbbasiert • Grafikkartenbasiert • Clientbasiert

• Movementbasiert (Bunny Hopping, Wallrunning) • Fix: Patches, statische Analysen	• Ghosting • Fix: Eingebaute Lösungen, Anti-Cheat	• Proxybasiert • Fix: eingebaute Lösungen, Anti-Cheat

Protocol Layer

Suppressed Update/Lag • Fix: Dead-Reckoning, Authorative Server predicts Movement	**Drophack/Disconnect** • Fix: Leave-Buster, Report-System • Problem: Spieler mit schlechtem Internet leiden	**Spoofing** • Sich als andere Spieler ausgeben • Fix: Authorative Serverm Encryption

Infrastruktur Layer

Information Exposure • Netzwerk oder Display-treiber modifizieren • Fix: On Demand Loading, Punk Buster	**Proxy** • Proxy zwischen Spieler und Server • Modifizieren der Commands zwischen Server und Client • Fix: Checksum	**Sensor Spoofing** • Modifizieren des Lokalisierungs-sensors • Fix: Travelspeed reduzieren, Verhaltens-analyse	**Communication Sniffer** • Fix: Encryption	**Emulation** • Verwendet für Automat-isierung • Fix: Encryption

Netzwerkarchitekturen

Client -> Server • Vorteile: Privatsphäre, geringer Traffic, robust • Nachteile: Serverkosten, Engpass serversseitig, lags	**Client on top of Server** • Deprecated • Vorteile: kein Server nötig • Nachteile: Mehr Arbeit für den Server, Engpass Serverseitig	**P2P** • Vorteile: Kein Server • Nachteile: schlechte Skalierung, schlechtes Anti-Cheat, Design-Probleme	**P2P-Hybrid** • Kombiniert P2P und Server-Client • Wichtige Daten an Server • Rest P2P

Server-Prinzipien

Authorative Server • Clients senden (Input) Daten an den Server • Server verarbeitet Daten und berechnet neuen Game State • Server sendet neuen Game State • Vorteile: Immer synchronisierter State, Cheating ist schwieriger • Nachteile: Lags beim warten auf Serverberechnung, Client Side Prediciton (used in FPS)	**Nicht-Authorative Server** • Client verarbeitet (Input) Daten • Client sagt Server Bescheid • Server synchronisiert • Vorteile: Einfacher zu implementieren, keine Client Side Prediction notwendig • Nachteile: Gehackte Clients können fürs cheaten verwendet werden, Timing Probleme

Netzwerkprobleme

Latenz • Zeit die ein Packet zum Ziel braucht • Ursprung: Entfernung, Serialisierung, Queing delays	**Jitter** • Variantion in der Latenz von einem Packet zu nächsten • Ursprung: Variable Weglängen, Variable Packet-Größen, Überlastung	**Packet Loss** • Inkonsistente Game States • Ursprung: Overflowing Queues, Link Layer Bit Errors, Routing Changes

Konsequenzen: Unnatürliche Inputs zwischen Input und Reaktion, Abgehackte Bewegung, Inkonsistenz

Latenzkompensation durch Game Engine

Netzwerkseitig	**Player Prediction**	**Opponent Prediction**	**Zeitmanipulation**
• Priorisierung von Packets • Last Mile Connection expandieren • Höhere Bitraten • Kein Einfluss auf diese Faktoren	• Client sagt die Serverantwort voraus und verbessert sie selbstständig	• Client schätzt die Position des Gegenspielers • Gegespieler schickt nur Updates bei Geschwindigkeits und Richtungs-änderungen • Client verwendet vorhergesagte Position wenn kein Update kommt • Reaktionsschneller aber inkonsistent	• Time-Delay: Server buffert uind schickt zuerst an die Spieler mit höchster Latenz • Time Warp

Andere Möglichkeiten: Delta Compression: Nur Updates senden, wenn Änderung vorhanden, Interest Management: nur Nachrichten an die Region des Clients, Update Aggregation
ABER IMMER TRAFFIC MINIMIEREN!!

Matchmaking

Ziel sind faire und gleichmäßige Spiele, Gleichmäßige Niveaumatches aber verhindern von langem Warten. Problem: Cheating/ Smurfing.

Elo System**:** Elo-Nummer R für jeden Spieler -> verwendet um gleches Niveau gegeneinander Spielen zu lassen. R wird zu R‘ geupdated mit der Formel $R'_A = R_A + k \cdot (S_A - E_A)$, wobei E_A die Gewinnwahrscheinlichkeit von Spieler A ist: $E_A = \frac{1}{1+10^{\frac{R_B - R_A}{400}}}$, $R_A$ist der aktuelle Elo-Wert für A und $R_B$der für Spieler B, k ist konstant und Abhängig von der Anzahl der gespielten Games (niedriger=mehr gespielte Spiele) und S_A ist das Spiel Ergebnis (1 für Win, 0.5 für Unentschieden und 0 für Verloren).

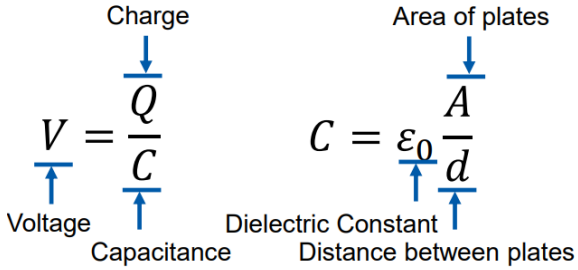
Sensoren

Sensoren sind ein Converter, der physische Werte mist und diese in Signale umwandelt, welcher von einem Observer oder Instrument geleswen werden kann.

Umgebungssensoren: Proximity Sensor (Näherungssensor), Barometrischer Drucksensor, Thermometer, Lichtsensor, Hygrometer (Feuchtigkeit), Touchsreen Sensor

Kamera: Sensor besteht aus vielen millionen Lichtsensoren, Bild machen funktioniert durch Photonen treffen auf Fotoseite das wird ein elektrisches Signal umgewandelt und bildet so ein monochromes Bild. Ein Farbfilter wird angewandt um farbige Bilder zu erhalten (üblicherweise Bayer-Filter).

Electret Mikrofon: Das Electret Mikrofon ist eine Art eines Kondensatormikrofons, es besteht aus einer Membran und einer Backplate. Eine dieser Beiden ist permanent Polarisiert. Die Bewegung in der Membran bedeutet eine Veränderung in der Distanz zwischen den Platten und dadurch eine Veränderung in der Spannung.



MEMS: MEMS (Mico-electro-mechanical system) haben einen integrierten Preamp und Analog-zu-Digital-Converter und bietet die selbe Funktion wie ein Electret Mikrofon, ist kleiner, günstiger und stabiler bei Feuchtigkeit und Temperatur im Vergleich zu Electret Mikrofonen.

Sensorqualität

Positive Eigenschaften	Negative Eigenschaften
• Sensitiv nur auf die gemessene Eigenschaft, also unabhängig von anderen äußeren Einflüssen • Output ist linear proportional zum gemessenen Wert	• Systematische Fehler (Drift), kann meist mit Kalibrierung behoben werden • Zufällige Fehler (Noise), kann meistens mit Signal Processing Techniken gefiltert werden

IMUs

Allgemein sind IMUs (Inertial Measurment Unit) ein Zusammenschluss aus mehreren Bewegungssensoren. Sie werden typischerweise in Smartphones, Gaming, Fitness Trackern und Fahrzeugen verwendet.

Beschleunigungssensor	Gyroskop	Magnetometer
• Misst Beschleunigungskraft, die auf das Gerät angewandt wird • Mikroelektromechanisches System • Gut in Langzeit (kein Drift) • Schlecht in Kurzzeit (Noise)	• Misst Rotationsgeschwindigkeit • Beeinflusst durch Temperatur • Additives Messrauschen	• Misst magnetische Feldstärke • Zeigt den Winkel zwischen dem absoluten Nordpol und der aktuellen Richtung • Beeinflusst durch Elektronik oder nahe Metalle • Hall-Effekt wird angewandt

GPS

Das GPS (Global Position System) verwendet Satellieten um die Position eines GPS Geräts zu trilaterieren durch die Distanz zu den Satellieten. Die Satellieten sind durch Atomuhren synchronisiert und schicken ein Signal in sychronen Intervallen ab indem die aktuelle Position und der Zeitpunkt enthalten ist. Der Empfänger ist passiv und erhält diese Signale, er errechnet die Time of flight (Distanz zum Satellit) und die Trilateration (Position).

Preprocessing

Ziel: Verwendbare Daten erhalten

Erwägungen: Potentiell notwendige preprocessing Schritte identifizieren: Recording Hard- & Software (Messbereich, Samplerate, Datenformat), Recording Setup (Anzahl Sensoren, Datentransfermethode), Späterer Nut zen (Normalisierung, Fixe Samplerate), Aufgenommene Werte (fehlende Werte).

Rohe Sensordaten sind meist nicht verwendbar, sie haben das Falsche Datenformat oder sind in der falschen Messeinheit, auch können sie Inkositent oder verrauscht sein. Daher werden folgende Schritte beim Preprocessen angewandt:

1. Konvertierung
2. Kalibrierung
3. Data Cleansing
4. Filtering
5. Frequncy Domain Analysis

Konvertierung: Rohe Bytestreams müssen in verwendbare Formate konvertiert und permanent gespeichert werden. Es ist daher sinnvoll den Bytestream zu unterteilen und in sinnvolle Einheiten umzuwandeln.

Kalibrierung: Geräte sind im Normalfall ab Werk kalibriert, müssen aber eventuell je nach Position und Zeit neu kalibriert werden. Die Orientierung kann mit einer Rotationsmatrix normalisiert werden.

Data Cleansing: Entfernen von fehlerhaften Daten, entsteht durch z.B. inkorrekte Formate, zu geringe Samplerate, inkonsistente Zeitstempel, fehlende Samples oder Werte außerhalb des angefordeten Wertebereichs. Nach vollständiger Aufnahmen kann man die Datzen Interpolieren oder Approximieren, kritische Daten können auch ganz entfernt werden oder falls das spätere Processing robust genug ist können die fehlerhaften Daten auch in diesem Schritt ignoriert werden. **Inkorrekte Zeitstempel** können durch fehlende Daten, unpräzise Zeitdaten auf dem Sensor oder unsynchrone Sensoren entstehen. Häufig können diese Daten ignoriert oder überschrieben werden. Eventuell müssen die Sensoren synchronisiert werden oder eine Fixed Samplerate verwendet werden.

Filtern: Rauschen in den Messdaten entfernen, folgende Filterarten werden meist dafpr verwendet:

- Einfach Filter: Mean Filter (1D), Box Filter (2D)
    - Box Filter: Filter Kernel mit 2D-Matrix, sagt aus wie benachbarte Pixel verbunden werden sollen und wird für u.a. Image postprocessing, motion blur, edge detection und smoothing verwendet.
- Erweiterte Filter: Hochpass, Tiefpass, Bandpass, Kalman-Filter

Frequency Domain Analysis: „Wie oft passiert etwas“ nicht „wann“, vorallem relevant für periodische Effekte wie die Samplerate. Bei der FDA wird die Amplitude für jede Frequenz angezeigt und das Zeitsignal wird mit einer Fourier Transformation in die Frequenz umgewandelt. Das **Nyquist Theorem** besagt, dass die digital korrekt dargestellte Frequenz kleiner als die Hälft der Samplerate in Hz sein muss. Häufig verwendete Filter: Highpass, Lowpass, Bandpass, Butterworth-Filter

Processing

Ziel: Wir haben Sensordaten und wollen Spezifische Infroamtionen.

Erwägungen: Welche Informationen beinhalten meine Daten, welche Limitierungen existieren und welche Informationen möchte ich aus meinen Daten erhalten?

Sensordaten werden dann in spezifische Informationen umgewandelt: **Regression** (Numerische Werte, Sampleratem Geschwindigkeit), **Klassifizierung** (In Gruppen, Aktivitätstypen, Objekttypen) und **Detection** (Sprung, Drehung, Ereignis). Daten sollten in eine Form gebracht werden, die für meine Anwendung ideal sind. Dafür verwendet man Fachwissen.

Feature Extraction: Beschreibt den Process um Features aus den gemessenen Daten zu generieren. Direktes umwandeln des Inputs in Daten, ersetzen dann die gemessenen Daten als Input für ein (ML) System. Typische Features einer IMU: Minimum, Maximum, Difference, Mean, Variance, Srandard Deviation, Skewness, Kurtosis, Body Height

- Informativ: Beinhaltet Informationen die zum relevanten Problem gehören
- Nicht-redudanz: Vermeide Features mit keinen relevanten Daten, sollten Unabhängig sein um Bloating zu vermeiden
- Komplexitätsreduktion: Meistens weniger Features als gemessene Werte

Algorithmische- und Regelbasierte Systeme: Simpler Ansatz bei dem man manuell eine Funktion schreibt, die die benötigten Informationen aus den Daten/Features produziert. Diese könnejn zum Beispiel Algorithmen oder Entscheidungsbäume sein und basieren immer auf Fachwissen und manueller Datenanalyse. Der Design und Implementations Aufwand ist stark abhängig von der Problemkomplexität. Typische Ansätze sind Thresholds und Entscheidungsbäume.

Machine learning Systeme:

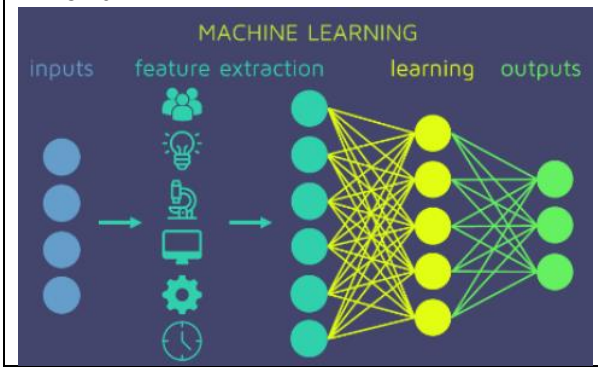
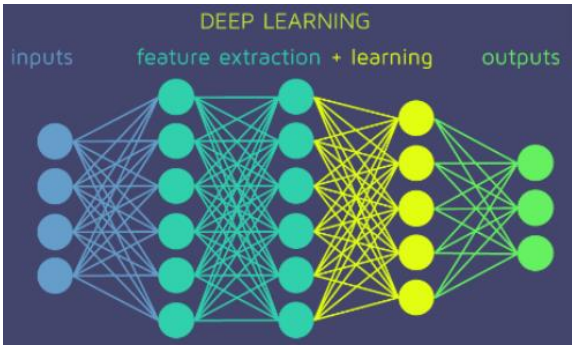
Automatisch lernendes Modell basierend auf den gemessenen Daten. Es gibt einige Dinge zu beachten. **Data Leakage** sollte vermieden werden: Daten die nicht im Datenset sein sollen, unbalancierte Datensets oder unnötige Daten.

Vorteile: Wenig Fachwissen notwendig, gute Ergebnisse bei komplexe Problemen und kein extra Designeffort Notwendig für neue Daten

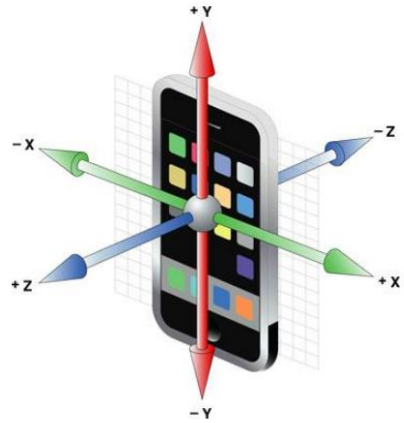
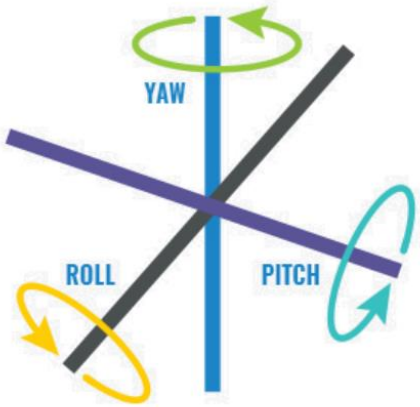
Nachteile: Man braucht viele representative Trainingsdaten, Fachwissen kann nur durch Features genutzt werden.

Feature-based learners:

- K-Nearest Neigbhor (kNN): Vorhersage basiert auf den k nächsten Nachbarn im Feature Space
- Random Forest (RF): Ensemble von Entscheidungsbäumen
- Support Vector machine (SVM): Versucht Samples zu separieren mit verschiedenen Klassen im Feature Space

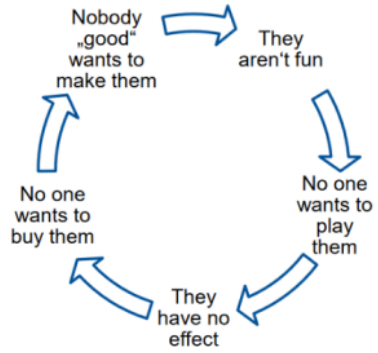
Shallow Learning	**Deep Learning**
• Manuelle Feature Extraktion • Einfach zu nutzen, out-of-the-box • Schnelles Training • Starke Ergebnisse wenn die Features gut sind	• Featureextraktion ist Teil des Lernprozesses • Braucht viel Adaption zu einem Problem • Braucht viel Trainingszeit • Starke Ergebnisse auch für komplexe Probleme
	

Achsen Accelerometer and Gyroscope

Accelerometer	Gyroscope
 $\tilde{a} = a^{(g)} + a^{(l)} + \eta \qquad \eta \sim N(0, \sigma_{acc}^2)$ acc measurement gravity acc linear acc Additive, zero-mean Gaussian noise	

	$\tilde{\omega} = \omega + b + \eta$ gyro measurement, angular velocity, bias, Additive, zero-mean Gaussian noise $\eta \sim N(0, \sigma_{gyr}^2)$ $\theta(t + \Delta t) = \theta(t) + \frac{\partial}{\partial t}\theta(t)\Delta t + \varepsilon$ angle at current time step, previous angle, gyro measurement, time step, approximation error $\varepsilon \sim O(\Delta t^2)$

Exergames

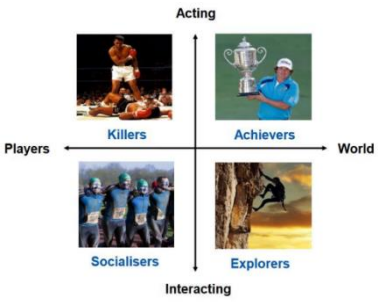


Exergames ist die Kombination aus Exercise und Games, Exergames verbinden physische Aktivität mit Spielen oder Spielelementen.

Es wird versucht Serious Games als **Motivator** um etwas zu tun zu verwenden, was man eigentlich lieber vermeiden möchte, dass kann **Lernen** (learning games), **physiche Aktivität** (Exergames) oder einfluss auf den **sozialen Stand** sein (social & political games).

Das Problem ist, dass Serious Games schlecht sind.

Exergames unterstehen aber auch einige **Herausforderungen**. Es ist wichtig, dass Spaßmachende Spielmechaniken implementiert werden und Andwendungen entwickelt werden, die oft und lange gespielt werden, sodass der Effekt auf den Spieler möglichst groß ist.



Bartler hat 4 Spielertypen definiert:

Mobile Exergames

Handyspiele können in drei Typen gruppiert werden:

- Casual /Arcade
- Hardcore -> Meist nicht für Smartphones ausgelegt
- Mobile-only -> Verwenden von Sensoren, mit dem Spieler und klein und leicht

Regular Game	Permeable Game	Pervasive Game
• Spiel versucht die echte Welt komplett zu separieren • Die meisten Brett & Video Spiele, LARPs	• Ein wenig gewollter Einfluss der echten Welt • Pay-to-win games, game administrators	• Spiele basierend auf Echtwelt Einflüssen • I Spy, location-based games

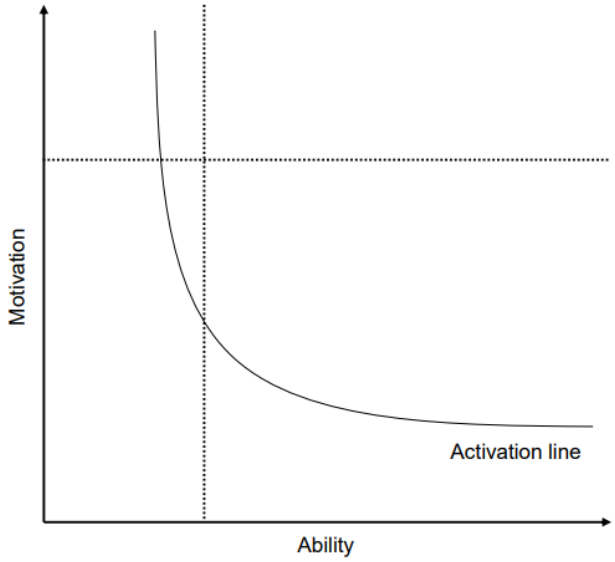
Context-Awareness: Es gibt drei Typen von Interaktionen: Signalbasier, Intervallbasiert und Eventbasiert. Trackers vs Games: Quantified Self (Fitnesstracker etc.), Gamified Anwendungen und Mobile Exergames.

Smartphone Historie:

- 1999: Blackeberry OS
- 2007: IPhone 1
- 2008: Apple App Store
- 2012: Android erreicht 75% Market Share
- 2013: Eine Milliarde Smartphones verkauft jedes Jahr

Smartphone Sensors: Kameras, Touch-Display, Brightness sensor, GPS/GLONASS Modul, IMU, Mikrofon, Proximity Sensor, Bluetooth, WiFi Module

Foggs Model



Fogg Behavior Model

Showing intended behavior (i.e., „something beneficial“) requires

- Motivation *m*
- Ability *a*
- Trigger *t*

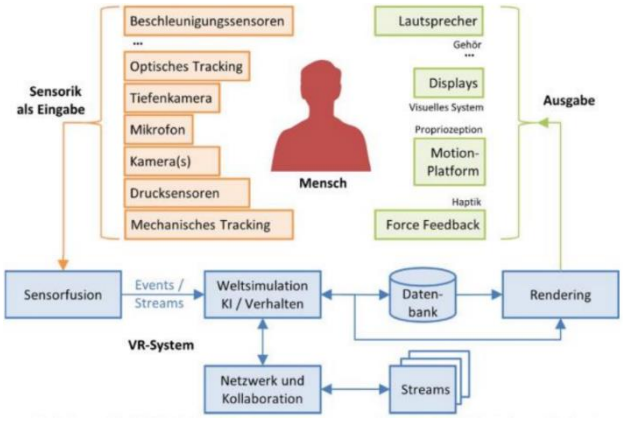
If I am the trigger by saying …
… I want you to move!
… outside – you should go running!
… run to the main train station!
… I'll give you money
… I'll give you 10.000 Euros

MET

Der MET (Metabolic Equivalent of Tasks) Wert bietet eine Schätzung wie groß der Energieverbrauch bei einer Aktivität ist. Die Standard Resting Metabolic Rate (SRMR) berechnet man MET = 1.0 kcal*kg-1*h-1. Der MET ist in drei Kategorien unterteilt: light < 3, moderate 3-6, vigorous > 6 und wird von Fitness Apps benutzt um den Kalorienverbrauch zu schätzen: kcal = MET*kg*h. Der **HRmax** Wert ist um das optimale Trainingsload zu berechnen. Hrmax = (220- Alter), Hrtarget = HRmax * 0.75.

Virtual Reality

Virtuelle Realität ist eine elektronische Simulation von Umgebungen, die mit einem HMD (Headmounted Display) und verkabelter Bekleidung dem Endnutzer eine Interaktion in realistischen 3-dimensionalen Situation ermöglichen.



VR vs AR vs MR: Alle kreieren immersive Umgebungen, der Unterschied ist, dass AR den Spieler nicht von der realen Umgebung isoliert (Video see-through) und MR ebenso (Optical see-through). AR ist meist auch teurer, ebenfalls muss der Scope betrachtet werden (AR Gaming?).

Anwendungsbereiche: Ingenieurswesen, klinische Anwendungen, psychologische Therapien, Training (Militär, Medizin), Automobilindustrie, Spiele

Terminology

Presence: Das Gefühl in einer Umgebung zu sein.

Telepresence: Presence in einer Umgebung mithilfe eines Kommunikationsmedium.

Virtual Reality: Eine reale oder simulierte Umgebung in welcher der Nutzer eine Telepresence erfährt.

Vividness: Der Repräsentationsreichtum einer vermittelten Umgebung, definiert durch formale Merkmale (Art und Weise) wie eine Umgebung Informationen für die Sinne darstellt.

Sensory Breadth: Anzahl der sensorischen Dimensionen, welche gleichzeitig präsentiert werden.

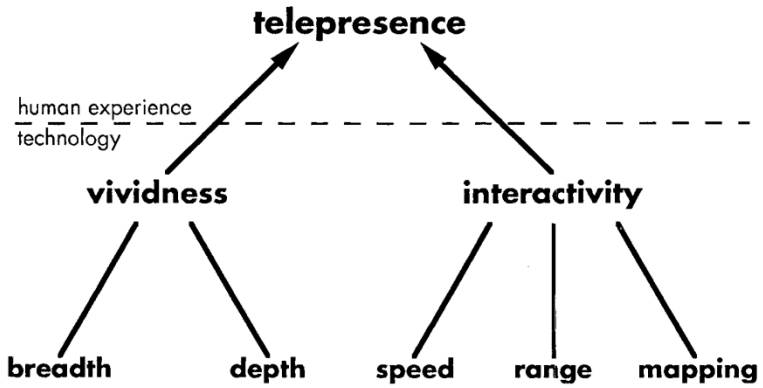
Sensory Depth: Die Auflösung in jeder diese Wahrnehmungssinne.

Interactivity: Ausmaß in welchem der Nutzer an der Veränderung der Form und des Inhalts der Umgebung partizipieren kann.

Speed: Rate, um Input in die virtuelle Umgebung zu überführen.

Range: Anzahl der Möglichkeiten für eine Aktion.

Mapping: Die Fähigkeit eines System, seine Steuerung auf natürliche und vorhersehbare Weise auf Veränderungen in der vermittelten Umgebung abzubilden.



Cybersickness

Vection: Die Wahrnehmung von eigener Bewegung.

Cybersickness: Visuell induzierte Motion Sickness, die in virtuellen Umgebungen auftritt. Führt zu Neusea aber nie zu erbrechen.

Symptoms

Nausea	**Oculomotor**	**Disorientation**
• Nausea • Increased Salivation • Sweating • Stomach Awareness • Burping	• Fatigue • Headache • Eyestrain • Blurry Vision • Difficulty Focusing	• Head fullness • Dizziness • Vertigo

Sensory Conflict Theory

Diskrepanzen zwischen körperlichen sensorischen Inputs verursachen Konflikte, die das Hirn nicht versteht.

- Visuell: Bewegungsrichtung, Geschwindigkeit, Beschleunigung
- Vestibular System: Bewegungsbeschleunigung
- Propioceptive System: actual posture and muscular effort

Faktoren für Cybersickness

Die Faktoren können in drei Kategorien aufgeteilt werden
Design:

- FOV: Großes FOV kann zu Cybersickness führen
- Beschleunigung: So kurz und wenig wie möglich halten

- Bewegunggeschwindigkeit: Bewegung in eine Richtung, während man in eine andere schaut kann Verwirrend sein. Drehungen und seitliche Bewegung sind besonders Problematisch
- Kameras: Rein oder raus Zoomen mit der Kamera kann zu Motion Sickness führen
- Positionstracking: Gerendertes Bild muss zu der physischen Bewegung passen

Technisch:

- Refreshing Rate: Zu geringe Refresh Rates führen zu Cybersickness
- Auflösung: Höhere Auflösung und Objektdichte führen zu Cybersickness
- Smearing: Pixel persistance durch eine endliche refresh Rate
- Strobing: Wahrnehmung mehrere Kopien des selben Bildes, durch endliche refresh Rate
- Judder: Smearing+Strobing, eine abgehackter, ungewollter Motion Blur

Persönlich

- Gesundheitszustand: Wenn man krank ist sind die Systome schlimmer
- Gender: Frauen sind anfälliger (geringer Unterschied)
- Adaptivität und vorherige Erfahrungen: Manche Nutzer adaptieren zu VR

Cybersickness reduzieren

- Physische Bewegung
- Keine Beschleunigung
- Konstante FPS
- Virtuelle Nase
- Kleineres FOV
- Latenz reduzieren

Motion Tracking

Full Body Tracking erhöht das Gefühl von Presence in VR.

Avatare sind Humanoid Avatare (realistische) um das Gefühl von Verkörperung zu erhöhen.

Die Position von echten Objekten verwendet **Inverse Kinematic** um Bewegung zu Rekonstruieren. Bei der Inversen Kinematic schließt man von der finalen Position auf die Knochenrotation. Das Problem ist, dass es mehrer mögliche Lösungen geben kann.

Markerless Optical Motion Capture Systems (Kinect)	Marker-based Optical Motion Capture Systems (Teslasuit, Controller)	Motion capture Systems using IMUs
• Günstig • Unpräzise • Okklusion hat negativen Einfluss	• Echtzeit • Hohe Genauigkeit • Teuer • Führt zu Unwohlsein	• Echtzeit • Leicht zu bedienen • Günstig • Drifting Probleme • Keine global Position

Immersionsgrad

Um die Immersion zu verbessern, kann man die verschiedenen menschlichen Sinnesorgane stimmulieren. Durch z.B. Visuellen und Auditiven Output, Haptische Handschuhe, Wind, Geruch und Geschmack.

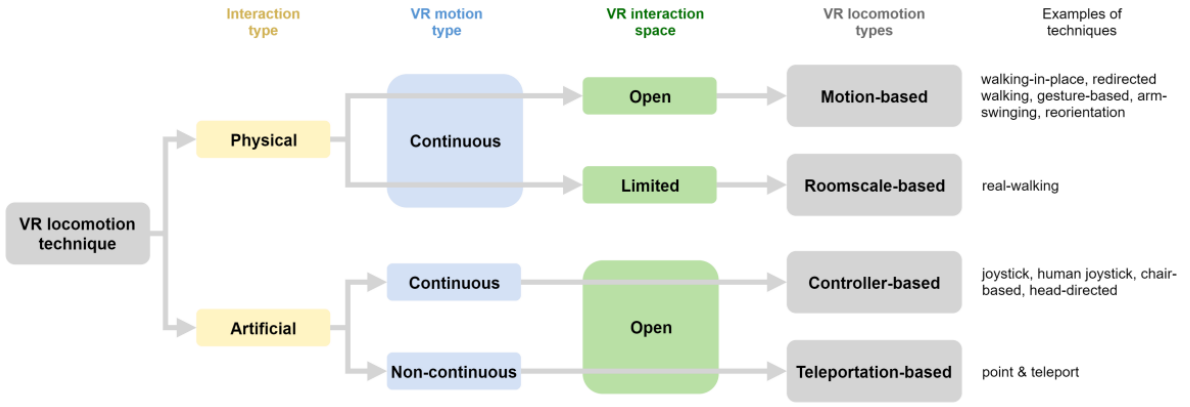
VR-Locomotion

Direkte Übersetzung von Bewegung in VR ist limitiert durch den physischen Tracking Space.

Folgende Techniken gibt es:

- Redirected Walking

- Walking in Place
- Steering (neigong) based
- Teleport
- Stationäre Geräte (Cyberwalk)

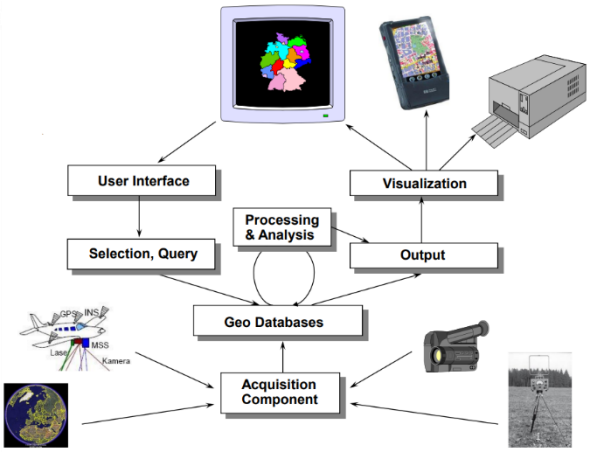


Geo

Als einen **digitalen Zwilling** beschreibt man z.B. einen Prozess, eines Produkts oder einer Dienstleistung, welches die reale und virtuelle Welt verbindet.

Geographic Information Systems (GIS)

Ein GIS ist ein raumbezigenes, rechnergestütztes Informationssystem, das aus Hardware, Software, Daten und Anwendungen besteht. Raumbezogene Daten können digital erfasst, gespeichert, reorganisiert und modelliert/analysiert werden. Diese Daten werden dann visuell Repräsentiert.



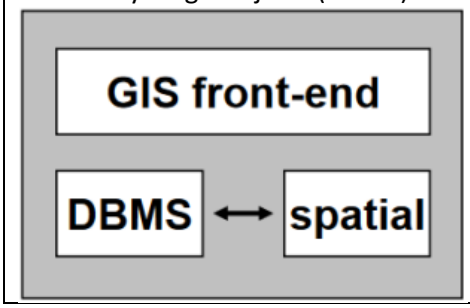
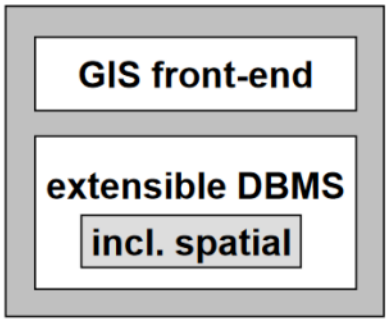
Zu den Anwendungsbereichen von GIS gehören Grasphisch-Interaktive Systeme, Visualisierung und Wissensgenerierung sowie GEO-Anwendungsbereiche (Statistik, Telekommunikation, Logistik, etc.)

Datentypen

Es gibt vier GIS Datentypen: **Vektordaten**, **Rasterdaten**, **Hybride Daten (Raster + Vektor)** und **Multimedia.**

Geodaten

Geodaten sind Daten über Objekte wie Landschaftsinformationen oder Infrastruktur auf der Erdoberfläche. Das essentielle Element ist hierfür die **Spatial Reference**. Diese beschreibt ein Objekt durch Position und topografische Informationen. Geodaten werden üblicherweise in Datenbanken, Datein, Rasterbildern oder Excel gespeichert. **Geodatenbanken** sind die häufigste Speicheranwendung, da sie ein effizientes verarbeiten der Daten erlaubt und ein interoperationäres Datenlager bieten. Es muss nur zwischen einem Relationalen und Objektorientierten Ansatz entschieden werden. Es gibt zwei Ansätze für Geodatenbanken:

Proprietärer Ansatz	**Objekt relationale Geodatenbank**
• Unterscheidung zwischen sachlcihen Informationen und Geometrie • Relationale DB + externe Daten (proprietäre Formate) • Binary Large Objects (BLOBs) 	• Objektrelationales Modell • Integrierter Speicher mit Geometrie • Erweiterung einer DB/DBS mit spatial Data 

Charakteristiken:

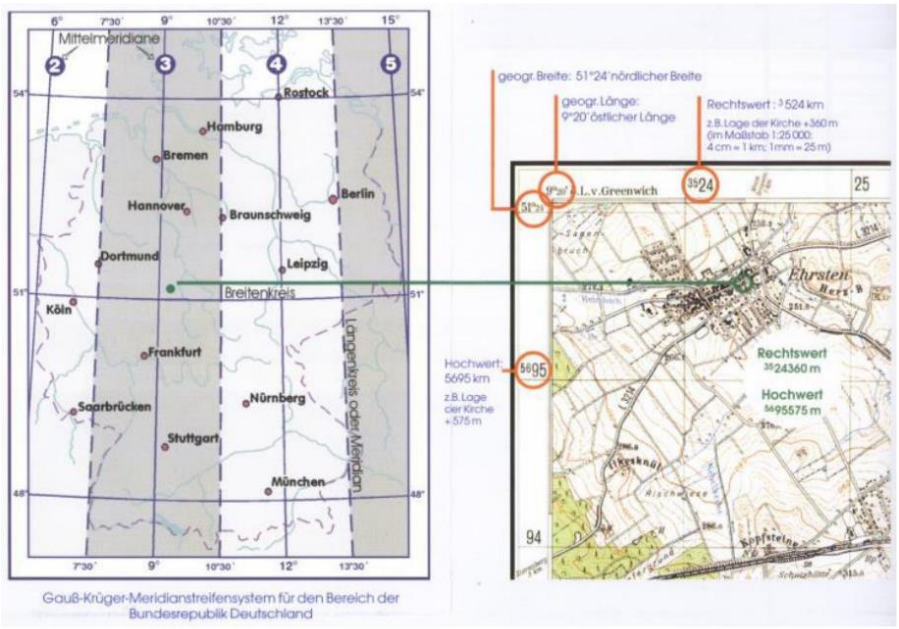
- Geometrisch (Position und Form, Kooridanten, Rasterdaten)
- Thematisch (Attribute, Visualisierung)
- Topografisch (Spatial Relationship)
- Temporal (Zeitpunkt)
- Metadaten (Daten über Daten)

Diese Geodaten können in unterschiedlichen **Dimensionen** sein:

- 0D (Punkt)
- 1D (Linie)
- 2D (Fläche)
- 2D + 1D (Objekt + Höhe)
- 2.5D (x,y + Höhe)
- 3D (Linien / Bereiche/ Volumen Modell)
- 4D (Raum + Zeit)

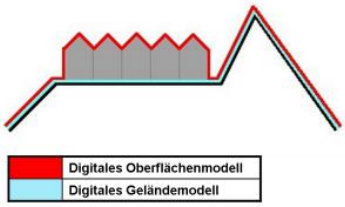
Koordinaten

Da die Erde keine richtige Kugel und nicht glatt ist, müssen wir wissen, wie wir Objekte auf dieser platzieren, häufig wird die Erde hierfür approximiert und ein Referenzsystem „über Normal Null“ verwendet. Um Die Erde auf Karten darzustellen muss sie in 2D projeziert werden. Hierfür verwendert man das **Gauß-Krüger Koordiantensystem.** Das Gauß-Krüger Koordinaten System basiert auf der Traversal Mercator Projektion, das Koordinatensystem der Projektion ist rechteckig und hat einen „Rechtswert“ und einen „Hochwert“. Der Zentralmeridian liegt in der Distanz von 3 Grad entfernt von Greenwich. Die Zentralmeridiane sind in östlcihe Richtung von Greenwich (0 meridian) numeriert.

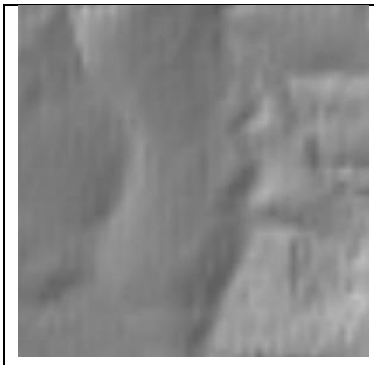
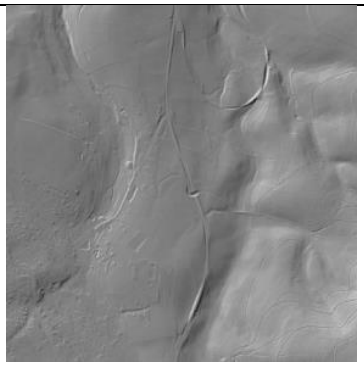
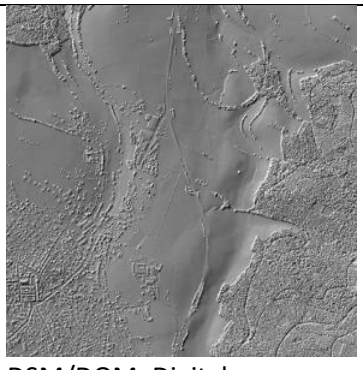


Digital Terrain Model

- **Digital Landscape Models** (DLM/DEM): DLM50, DLM250, DLM1000
  - Erstellt mithilfe photogramatischer Analyse von Orthophotos
  - DGM25 mit Höhenauflösung 2-3m, DGM5 mit Höhenauflösung 0,3-0,5m
  - Basis bieten Digital gereferenced 4 kanal Orhtophotos mit 20cm **Ground Sampling Distance** (GSD)
  - Unterschied zwischen DLM und DOM/DSM siehe Bild

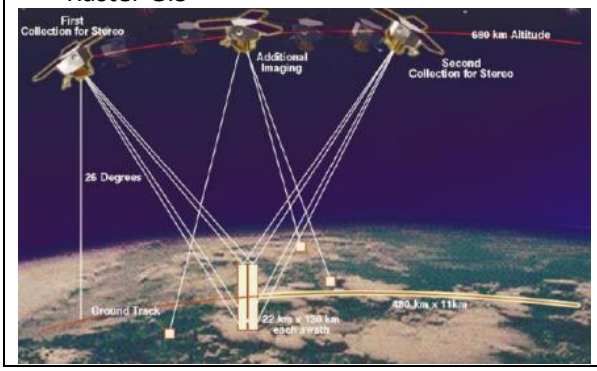


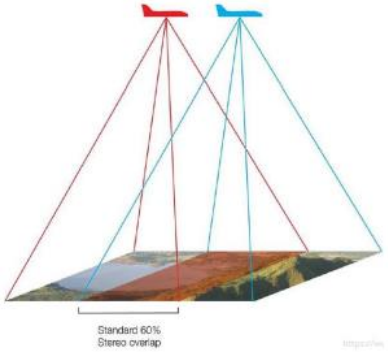
- **Digital Terrain Models** (DTM/DGM): Raster mit 1m (DGM1), 2m (DGM2), DGM5, DGM10, DGM25, DGM50, DGM 200, DGM1000
- **Digital Topographic Maps** (DTK): Maßstab 1:10000 (DTK10), DTK25, DTK50, DTK100,...
- **Digital Orthopohotos**: 20cm Auflösung (DOP20), 40cm Auflösung (DOP40)

 DHM40: Digitales Höhenprofil, Fotogramatisch	 DEM/DGM: Digitales Geländemodell, Laser, Dotspacing 1-3m	 DSM/DOM: Digitales Oberflächenmodell, Laser, Dotspacing 1-3m

Photogrammetry vs remote sensing

Remote Sensing	Photogrammetry
• Landsat, 30m Auflösung • Sattelieten und Flugzeuge • Feldmapping, Landuse • Automatische Klassifizierung • Raster GIS	• Orthophotos, 30cm Auflösung • Flugzeuge • Geometrie, Position,Größe, Form • Automatische Bildanalyse • Vector GIS





Offizielle Geodatenbanken

Enthalten Daten über Objekte, Landschaftsinformationen, Infrastruktur auf der Erde.

Das AAA-Modell hat folgende Datenbanken:

- ALKIS: Land survey register
- ALK: automated cadastral map
- AFIS: Anchor point information system
- ATKIS: official topographic-cartographic information system