Pink Trenchcoat

a cyberpunk rule-set

Serbitar

July 13, 2023

Contents

Chapter 1

Basics

This chapter will cover the basics of Pink Trenchcoat including standard RPG nomenclature as wells as methods of conflict resolution. The rule system uses a fixed set of resolution methods, which are covered here, that will be used throughout the system exclusively.

1.1 Definitions

A couple of basic descriptions and definitions are given here. Throughout the book everything that is a game term with a defined meaning in the game is written in *italics*, and in upper-case if it is a noun. All game terms should be found in the index at the end of the book.

1.1.1 Gamers

Everyone that is taking part in the game is a *Gamer*.

Game Master The *Game Master* is the person that is not playing their own *Character*, but all the *Characters* that are not being played by a *Player*.

Players A Player is a *Gamer* that is only playing their *Character* and maybe *Characters* that are closely connected to this *Character* like *Drones*, *Agents* or *Contacts*.

1.1.2 Characters

A *Character* is an entity that can actively make decisions in the game world and act on those decisions. In Pink Trenchcoat this includes (Meta)-Humans, but also *Agents*, *Drones*, *Spirits* and more.

Player Characters A *Player Characters* or *PC* is a *Character* that is directly and often exclusively controlled by a *Player*.

Non-Player Characters All *Non-Player Characters* or *NPC* are most often controlled by the *Game Master*.

1.1.3 Mathematics

Pink Trenchcoat's resolution system only uses integers. Although during calculation a number mit be not an integer, it needs to be rounded to the next integer for any kind of *Test*.

Rounding Fractions are always rounded mathematically correct. This means that 0.5 is rounded to 1.

1.2 Dice

Like most game systems Pink Trenchcoat uses dice to act as a randomizer for *Tests*. This is done to increase tension during the game session and include a random element so that players can not plan everything in advance with 100% certainty. However, if the gaming group so chooses, the rule set can be used completely without dice, as the average result of a die roll is always 0.

Pink Trenchcoat uses five six-sided dice with two "-", two blank and two "+" symbols also known as FUDGE dice. They are always used together and there are no other dice rolls used.

Almost always a player will roll only 5 dice, and the game master will secretly roll the other 5 dice, either because its an *Opposed Test*, and the game master is performing the roll for the opposition, or because it is not an *Opposed Test* and the game master will roll 5 dice because the player should not be sure of the outcome. Only in cases where the player is managing the situation fully they should roll the full 10 dice, but either roll 5 dice twice or use differently coloured dice to calculate *Criticals* and other functionality the dice roll is covering.

Every test requires 10 dice to be rolled in total.

In this rule set, 5 FUDGE dice will always be referred to as:

5f

while the full 10 FUDGE dice will always be referred to as:

10f

1.2.1 Result

The Result of *10f* is calculated by rolling 2 times 5 dice and summing all "+" as 1 and all "-" as -1 while blanks count as 0.

If the Result of a *10f* roll needs to be calculated in this rule system it will be denoted as:

10fR

Probability Distribution The average *Result* of any dice roll in Pink Trenchcoat is always 0. The number of total dice rolled is also always 10 (although, sometimes, the dice are rolled by different people for psychological reasons, mathematically this makes no difference).

Using 10 dice, the following statistics apply the outcome of *10fR*.

Probability for exactly rolling a value Sometimes it is good to know what the probabilities to exactly roll a value are. The probability distribution of the *10fR* is a gaussian with mean of 0 and a standard deviation of about 2.6.

Figure 1.1: *10fR* Probability Distribution

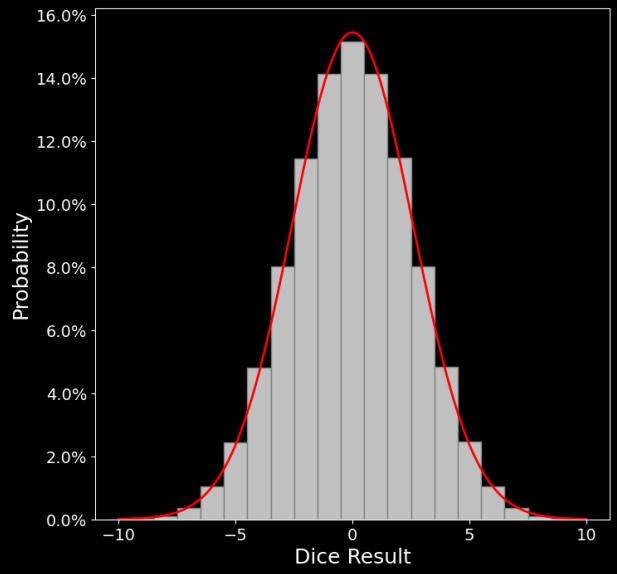

Table 1.1: 10fR Probabilities

Roll exactly	Chance	one in
-10/10	0.0014%	71000
-9/9	0.016%	6100
-8/8	0.088%	1100
-7/7	0.36%	280
-6/6	1.0%	96
-5/5	2.4%	41
-4/4	4.8%	21
-3/3	8.0%	13
-2/2	12%	8.7
-1/1	14%	7.1
0	15%	6.6

Probability for rolling a value and lower/higher Most of the time it is important to know the probability to at at least a certain number or higher, or the inverse, the chance to roll a certain number or lower. Both are important to judge if a *Test* will fail or succeed.

As a rule of thumb, rolling below -5 or above 5 is not happening often. This also means that *Tests* that only fail when a value smaller than -5 is rolled should only be done if the success or how well it succeeded or failed is critical for the game. Instead it can just be assumed that the *Test* succeeded normally.

Figure 1.2: *10fR* Cumulative Probability Distribution

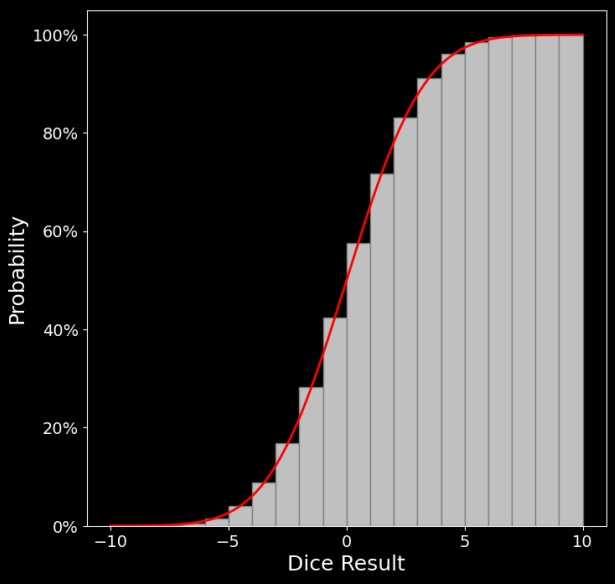

Table 1.2: 10fR Cumulative Probabilities

Roll exactly or		Chance	one in
bigger	smaller		
10	-10	0.0014%	71000
9	-9	0.08%	5600
8	-8	0.11%	940
7	-7	0.46%	220
6	-6	1.5&	66
5	-5	3.9%	25
4	-4	8.8%	11
3	-3	17%	6.0
2	-2	28%	3.5
1	-1	42%	2.4
0	0	58%	1.7

1.2.2 Anomalies and Criticals

The *Result* is not the only quantity that the dice deliver. Another one is Anomalies and Criticals. They are in principle the same thing, but Criticals are much more seldom and extreme in their effect.

Criticals and Anomalies are determined only looking at the *5f* roll of either the player and the game master. This means that both parties in an *Opposed Test* can generate a Critical or Anomaly at the same time. They happen if multiple dice show similar symbols.

Anomaly To determine Anomalies the number of similar symbols have to be counted. Every time 4 dice of a *5f* roll show the same symbol, an Anomaly happened. This can be four "+" (Positive Anomaly), four "-" (Negative Anomaly) or four blanks (Neutral Anomaly).

The chance to roll an Anomaly is 4.1% for any kind of Anomaly. This means that the chance is 12.3% to have any kind of Anomaly in a *Test*. The Game Master needs to decide whether they want to ignore *Anomalies* in an *Opposing Test* if the opposing faction is an NPC. The same applies for the other *5f* that are rolled in a *Unopposed Test*.

Positive and Negative Anomaly The result of a positive or negative Anomaly enhances the outcome of the *Test* in a positive or negative way respectively, but does not change the *Result*. The Game Master needs to look at the situation and think of any positive or negative effects that could happen.

This includes:

- Taking more/less time of an action in combat that normally can not be slowed/sped up
- getting into a advantageous/disadvantageous position when performing a melee attack
- increasing/decreasing connection status of a contact when doing legwork
- using less/more resources when crafting an item

Neutral Anomaly A neutral Moderate Critical should just create unusual side effects to an outcome. Again the Game Master should be free to invent anything coming to their mind.

For example:

- A
- b
- c

Critical Criticals happens if all 5 dice of a *5f* show the same symbol. As with Anomalies there are positive, negative and neutral Criticals. Both the chance and the effect of a Critical are much more radical than an Anomaly.

The chance to roll any kind of Critical is 0.4%.

Positive Critical If there is a remote chance of the *Test* succeeding, it will. This does not allow *PC* to do things that are impossible like surviving an atomic blast or succeeding in a wrestling match with a dragon, but anything close to that.

Negative Critical The *Test* fails and it fails spectacularly. The Game Master is free to invent any convenient explanations. There is always a way something can fail.

Neutral Critical The *Result* of the *Test* is not affected, but something very strange happens. The Game Master can do whatever they see fit.

1.2.3 Non Blanks

The Non Blanks of *5f* is calculated by counting all the "+" and "-" symbols, resulting in a number from 0 to 5.

If the Non Blanks need to be calculated from a *Test* this is denoted as:

5fN

Note that does not mean that an additional *5f* need to be rolled in addition to the *10f* of the *Test* itself, but instead use the *5f* from the existing *10f* roll.

The Non Blanks are used for various secondary purposes of a dice roll.

Figure 1.3: *5fN* Probability Distribution

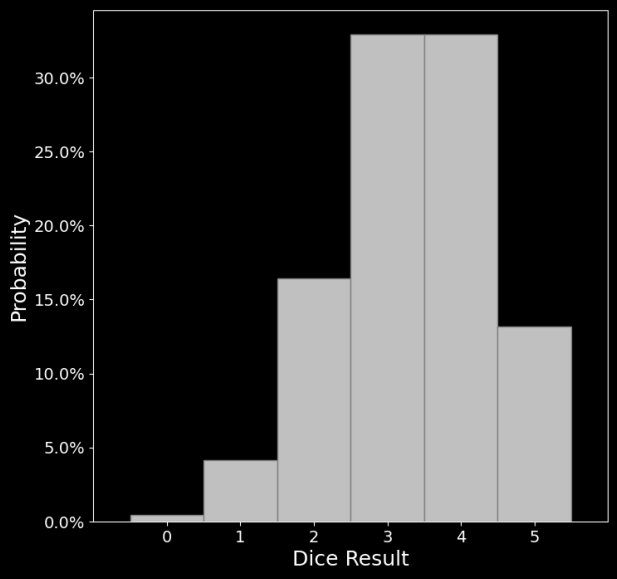

Table 1.3: 5dN Probabilities

Roll exactly	Chance	one in
5	13%	7.6
4	33%	3.0
3	33%	3.0
2	16%	6.1
1	4.1%	24
0	0.4%	240

1.3 Tests

A test determines the outcome of a certain action, which has a certain probability to fail and which has an important impact on the game session if it fails. Tests should not be rolled if it is clear that the test will succeed, like in the case of opening a door. Tests should also not be rolled if the result is irrelevant for the game session, like when a character is trying to beat a popular game in their spare time.

Every time the outcome of an action is, given the capabilities of the acting character, in doubt, or if the result needs to be quantified, a *Test* is rolled.

1.3.1 Test Anatomy

All Tests in Pink Trenchcoat look like the following:

$$\begin{aligned} \textit{Test Quality} = \textit{10fR} + \textit{Ability Score(s)} \\ + \textit{Modifiers(s)} \end{aligned} \tag{1.1}$$

The *10fR* was already explained in the previous section.

Ability Score The Ability Score is a number giving the proficiency of the person or entity that performs the *Test* to achieve the result. The higher, the better.

Normally Ability Scores are either *Attributes* or *Skills* of a character.

Limits Sometimes tools and other situational effects are not modeled as a *Modifier* that is added or subtracted but as a *Limit* to the *Ability Score*. In case the *Ability Score* can not be higher than the *Limit*.

Limits to the *Ability Score* are noted as follows:

$$Ability\ Score(Limit) \tag{1.2}$$

Modifier *Modifier* can be anything from a threshold that needs to be achieved to circumstantial *Modifiers* like visual conditions, tools or wounds that can change the result of a *Test*. If a *Modifier* is helping the *Character* performing the *Test*, like good tools, or support from friends, it is positive. If it is an obstacle of problem for the *Character* performing the *Test*, the *Modifier* is negative.

Test Quality The *Test Quality* ot *TQ* is the value that results from adding the *10fR* the *Ability Score* and the *Modifiers*. If the *Test Quality* is zero or positive, the *Test* succeed, if it negative it failed. The higher the *Test Quality* the better the result and the lower the *Test Quality* the worse the failure.

Table 1.4: Test Quality

TQ	Description
< -9	Epic Fail
-7 to -9	Severe Failure
-4 to -6	Decisive Failure
-1 to -3	Failure
0	Barely made it
1 to 3	Acceptable
4 to 6	Good Result
7 to 9	Exceptional
> 9	Epic Success

1.3.2 Unopposed Tests

In an *Unopposed Tests* a *Character* is not testing against another *Character* but against the environment. Typical *Unopposed Tests* include:

- crafting something
- climbing a wall
- running fast
- remembering something

In this case, the *Ability Score* is just the relevant value from the *Character* and the *Modifier* is the difficulty of the task plus any additional situational *Modifiers*.

This rule system defines the *Ability Scores* to use in an *Unopposed Test* in the following notation:

$$Ability\ Score_{Acting\ Character} + Modifier \tag{1.3}$$

In case of a climbing test for a given wall, that would be:

$$Climbing - 6$$

1.3.3 Opposed Tests

If two *Characters* are fighting against each other, either literally in melee combat or figuratively when one *Character* tries to sneak by and the other to spot the sneaker, an *Opposed Test* is called for. In this case, both involved *Characters Ability Scores* are used. The definition of the *Test* explains which values of a *Character* are used, as this can be the same, in the case of melee combat or be different in the case of sneaking.

This rule system defines the *Ability Scores* to use in an *Opposed Test* in the following notation:

$$Ability\ Score_{Attacker}\ vs.\ Ability\ Score_{Defender} \tag{1.4}$$

In case of melee combat this would mean:

$$Melee\ Combat\ vs.\ Melee\ Combat$$

In case of sneaking it would mean:

$$Stealth\ vs.\ Perception$$

The final *Test Quality* is then calculated as follows:

$$\begin{aligned} Test\ Quality = 10fR \\ &+ Ability\ Score_{Attacker} \\ &+ Modifiers(s)_{Attacker} \\ &- Ability\ Score_{Defender} \\ &- Modifiers(s)_{Defender} \end{aligned} \tag{1.5}$$

Ties are broken either by bespoke tie breakers given in the specific rules section, or, and if those tie breakers still end in a tie, by the*Game Master*. They can decide to either flip a coin or decide for themselves.

1.3.4 Supported Tests

If one or more *Characters* are helping another *Character* to do a task that can not be split into subtasks, but all characters have to do the full task, this is a *Supported Test*.

- climbing a wall together
- helping a character to sneak
- crossing a mine-field

In this case the *Ability Score* for the *Supported Test* is the average *Ability Score* of all the *Characters* involved. The *Modifiers* for the *Supported Test* are the average *Modifiers* of all the *Characters* involved -1.

The *Game Master* decides which *Tests* can be supported.

1.3.5 Collaborative Tests

If one or more *Characters* are working together, distributing the work to perform a task that can be broken down into independent parts this is a *Team-Play Test*. The goal is to either increase the quality of the result, or to speed up the process by using less *Task Time*.

- crafting an item

- collecting information
- repairing a vehicle
- summoning a spirit

In this case the *Ability Score* for the *CollaborativeTest* is the average *Ability Score* of all the *Characters* involved. The *Modifiers* for the *Collaborative Test* are the average *Modifiers* of all the *Characters* with an additional benefit depending on the number of *Characters* working in the *Test*.

Table 1.5: Collaborative Test

Characters	Modifier
3	+1
10	+2
100	+3
1000	+4

The *Game Master* decides which *Tests* can be *Collaborative Tests*.

1.3.6 Task Time

In most *Tests* a *Character* can spend more ore less *Task Time* to do the task better or achieve an outcome faster. In the case of spending more *Task Time*, this will either make a success possible or allow for a better result.

Table 1.6: Extra Time

Time Multiplier	Modifier
x0.5	-6
x0.7	-3
x3	+1
x10	+2
x100	+3
x1000	+4

If not explicitly allowed or disallowed by the rules the *Game Master* decides whether spending more or less *Task Time* is possible.

1.4 Thoughts and Philosophy

The main idea behind the use of 10f is are th follows:

- easy calculable expectation value
- possibility of suspense
- possibly only one roll per decision

Expectations It should be easy to calculate the average and preferably also the most common outcome of a *Test*. In this rule-system both are the same and are extremely easy to judge, being zero. The average outcome of a *Test* is thus always the sum of *Ability Score* and *Modifiers*.

It is also important that the average outcome is occurring more often than the extremes. This is not the case in linear systems like d20 where the extremes of 1 and the 20 are just as probable as the average 10 or 11. This is important because although a lot of tings are handled by rules, the vast majority of expectations about a game come from the real world, which mostly follows gaussian-like distributions and thus shape player expectations.

Suspense

One Roll This is not only design concept, but should also be the philosophy of each *Game Master*. If there is no decision between two *Tests*, the second test is unnecessary and should be avoided. A big *Test* should only be divided into smaller more granular *Tests* it is both interesting for the *Players* and there are decisions between the *Tests*. Otherwise just one general Test should be made.

Chapter 2

Character

This chapter describes *Characters*. Currently this chapter describes only meta-human *Characters* with a physical body to be played by a *Player*. In principle, certain types of *Agents* and *Spirits* could also be played, but are currently not in scope of this rule-set. body in particular.

2.1 Attributes

Attributes are very central values in defining a *Character's* abilities. They give a broad description of a *Character's* strengths and weaknesses and are influencing both final *Skill* values as well as derived *Characteristics*.

The base value for most *Attributes* of an average human is 8.

Table 2.1: Attribute Values

Value	Description
≤ 3	Disabled
4–5	Challenged
6–7	Underdeveloped
8	Average
9–10	Improved
11–12	Superior
13–14	Exemplar
≥ 15	Superhuman

Attribute values in Pink Trenchcoat are logarithmic with a base of 3. This means that a *Character* with *Strength* 11 is twice as strong as a *Character* with *Strength* 8 which in turn is twice as strong as a *Character* with *Strength* 5. This fact is only influencing certain *Characteristics* like *Carrying Capacity* and does not need to be kept in mind in most situations.

2.1.1 Mental Attributes

Pink Trenchcoat uses four *Mental Attributes*.

Charisma *Charisma* describes a *Character's* ability to positively affect other people in interactions. Highly charismatic people instantly get the attention of others, are often favored, and respected. A person with a low *Charisma* value is often ignored and sometimes not taken seriously. *Charisma* is also required to connect with people emotionally and understand emotional context of a conversation.

Ware is negatively affecting *Charisma* as it detaches the *Character* from itself.

Inutition *Intuition* describes the *Character's* ability to intuitively ans subconsciously process information. It describes not how fast or how much the *Character* can process, but how well. Furthermore a high *Intuition* value helps the *Character* to grasp a situation faster and perceive better.

Logic *Logic* describes the raw processing power and storage capacity of a *Characters* brain. Combined with *Intuition*, both attributes form the *Characters* IQ. A high *Logic* value helps with most *Craftsmanship* and all *Knowledge Skills*.

Willpower *Willpower* represents the amount of control the *Character* has about their mind and body. How far they can force their body to go, and how well to withstand temptations of any kind. It is also a measure for courage.

2.1.2 Physical Attributes

Pink Trenchcoat uses four *Physical Attributes*.

Agility *Agility* represents a *Character's* nimbleness and dexterity. The motions of a *Character* with high *Agility* look fluid and smooth, while low *Agility* motions look stocky. *Agility* is important for all *Close Combat* and most *Physical Skills*. Larger *Characters* normally have lower *Agility*.

Body Body describes a *Character's* ability to endure physical strain, and keep going, even when exhausted. It also influences how much *Damage* the body can take before collapsing. *Body* is independent of *Size*, meaning that a large *Character* does have the same average *Body* as a smaller one.

Coordination *Coordination* is the ability to control your body the way you want, especially hand-eye coordination. Although a *Character's* body can be very agile, as long as the *character* can not control it in the right way, it may not help much. *Coordination* is important where the *Character* works with his hands, like in *Ranged Combat* or most *Craftsmanship Skills*.

Strength *Strength* measures the raw power of a *Character's* body, the pure muscle volume. Most *Physical Skills* benefit from a high *Strength* value. *Strength* generally increases with *Size*.

2.1.3 Other Attributes

Fate *Fate* is a measure of a *Character's* luck, the favour of the gods or their balance score with the universe itself. Or it is just a gamistic resource that can affect *Tests*.

Fate refreshes every game session and can be used to modify *Test* either before or after the roll. It can only be taken in high stake moments, that are critical for the story or the *Character*. The *Game Master* decides if this is the case.

Optional: Anomaly Ownership A *Player* can also, by spending *Fate*, take ownership of an *Anomaly* they have rolled. This means that now the *Player* instead of the *Game Master* decides and describes what the special effects of the *Anomaly* are. The *Game Master* however needs to accept the effect and decides how much *Fate* it costs.

Table 2.2: Fate Costs

Value	Description
1	+1 before
3	+2 before
6	+3 before
4	+1 after
9	+2 after
1+	own *Anomaly*

Table 2.3: Skill Values

Value	Description
0	Untrained
1–3	Amateur
4–6	Journeyman
7–9	Senior Journeyman
10–12	Master
13–15	Elite
16–18	Legendary
≥ 19	Godlike

Magic

Size

2.2 Characteristics

2.2.1 Health

Life

Wound Limit

Damage Pip

Wound Heal Time

2.2.2 Athletics

Carrying Capacity

Combat Speed

Action Costs Action Costs describe how many *Ticks* a *Character* needs to spend for any kind of *Action*.

Reaction *Reaction* determines the speed a *Character* can react to external events like suddenly appearing dangers or actions in combat.

2.3 Skills

2.3.1 Combat

2.3.1.1 Melee Combat

2.3.1.2 Direct Weapons

2.3.1.3 Ballistic Weapons

2.3.2 Craftsmanship

2.3.2.1 Chemistry

2.3.2.2 Mechanics

2.3.2.3 Mechatronics

2.3.2.4 Medicae

2.3.2.5 Social Crafting

2.3.3 Empathy

2.3.3.1 Discourse

2.3.3.2 Influence

2.3.3.3 Etiquette

2.3.3.4 Scrutiny

2.3.4 Magic

2.3.4.1 Assensing

2.3.4.2 Enchantment

2.3.4.3 Evocation

2.3.4.4 Invocation

2.3.4.5 Physical

2.3.4.6 Acrobatics

2.3.4.7 Athletics

2.3.4.8 Sleight of Hand

2.3.4.9 Stealth

2.3.5 Piloting

2.3.5.1 Anthroforms

2.3.5.2 Gunnery

2.3.5.3 Pilot Air

2.3.5.4 Pilot Ground

2.3.6 Processing

Cracking

2.3.6.1 Navigation

2.3.6.2 Perception

2.3.6.3 Software

2.3.7 Resistance

2.3.7.1 Composure

2.3.7.2 Interaction

2.3.7.3 Memory

2.3.7.4 Survival

Chapter 3

Combat

3.1 Timing

3.1.1 Resolution Order

In Pink Trenchcoat combat is resolved by a continuous, always increasing time value.

Tick This time value is measured in *Ticks*. A *Tick* is a time measure of about 0.3 seconds.

Current Tick Combat continuously advances the *Current Tick* that represents the the current point in time.

Initiative Score The emphInitiative Score represents the *Current Tick* in which a *Character* can *declare* and take *Actions*. Ties in *Initiative Score* are broken by the *Characters Reflex* value.

Phase A *Phase* lasts 20 *Ticks*. Each Phase, this means on *Current Tick* 20,40,80 and so on *Continuous Effects* like fire, toxin damage and bleeding are resolved.

Combat Flow Characters who's *Initiative Score* matches the *Current Tick* are allowed to *declare* an *Action*. An *Action* that would increase a *Character's Initiative Score* to greater than the

$$\textit{Maximum Initiative Score} = \textit{Current Tick} + 20 \quad (3.1)$$

can not be declared.

After *declaring* an *Action* the *Characters Initiative Score* is immediately increased by a value depending on the *Action*. If no *Interrupts* occur, the *Action* is *resolved*.

When all eligible *Characters* have taken their *Actions*, the *Current Tick* is advanced to the next meaningful value which is normally either the next lowest *Initiative Score* of a *Character*, the next *Phase* if there are any *Continuous Effects* to *resolve* or the *Tick* a *Character* in *Delay* wants to *act*.

Interrupt Instead or in addition to acting on their *Initiative Score*, a *Character* can also chose to *interrupt* another *Character* after *declaring* an *Action*. After a *Character* declared their *Action* and increased their *Initiative Score*, but before it was *resolved*, the *interrupting Character* can *declare* their *interrupting Action* (and also immediately increase their *Initiative Score*). The *Interrupting Action* can itself be interrupted by a *Character* that has not yet declared an *Action* this *Tick*.

A *Reflex Test* decides the order in which *Characters* taking part in the *Interrupt* are *resolving* their *Actions*. Each *Character* receives a *Modifier* equal to:

$$\textit{Interrupt Modifier} = \textit{Current Tick} - \textit{Initiative Score} \quad (3.2)$$

In addition, the following *Modifiers* apply:

Table 3.1: Interrupt Modifiers

Modifier	Situation
+9	*Overwatch Action* triggered
+3	*Aiming* or *Watching Character*
-3	per level of *Interruption*

Actions are *resolved* starting from the highest *Test Quality* to the lowest. Ties are broken first by lowest *Initiative Score*, then by *Reflex* value.

3.1.2 Starting Combat

Once the Game Master decides that the Game should transition from *Narrative Time* to *Combat Time*, the *Characters* starting *Initiative Score* needs to be determined. To do so, each Character rolls a *Reflex Test*. The negative *Test Quality* determines the initial *Initiative Score*.

Apply the following *Modifiers*:

Table 3.2: Surprise Modifiers

Modifier	Character State
+6	Initiated first *Action*
0	Actively anticipating combat
-3	Suspicious
-6	Not expecting combat
-9	Deeply involved

3.2 Combat Actions

3.2.1 Action Times

To reduce cognitive load, *Action Times* are separated into four categories. Depending on the *Character* and their *Attribute*values, these *Actions* take a different amount of *Ticks* to fulfil.

Free Action *Free Actions* are the shortest kind of *Action*. They require no or almost no thoughts and can be executed almost immediately.

Simple Action A *Simple Action* can still be done quickly, or be triggered by reflex, but is not instant.

Half Action *Half Actions* are the standard basic Action.

Full Action *Actions* taking a lot of concentration or a lot of time.

3.2.2 Meta-Actions

Meta Actions are different compared to normal Actions in a sense that they do not follow the typical scheme of Declaration, Initiative Score increase and Resolution.

Delay A *delaying* Character does not have a *Initiative Score*, but can chose to *act* at any future *Current Tick* and thus immediately gets the *Current Tick* assigned as *Initiative Score*, before *declaring* the *Action*. They have to do so before any other *Character declared* an *Action* in that *Tick*. If another *Character* already *declared* an *Action*, the *delaying Character* can still chose to *act*, but will need to perform an *Interrupt*.

After performing any *Action*, *delay* ends.

Watch When *declaring Watch* a *Character* has to chose a suitable *Character* or object to *watch*. The *Character* immediately loses their *Initiative Score*. The watching Character is granted a bonus of

A *Character* can switch from *Watch* to *Delay* at any time.

Overwatch When *declaring Overwatch* a *Character* has to chose both a *specific Condition* and a *specific Action*. The *Character* immediately loses their *Initiative Score*.

Once the *specific Condition* is fulfilled, even if in *declaration* the *Character* immediately performs the *specific Action*. The *Character* is assigned an *Initiative Score* according to the *Current Tick* plus the value depending on the *Action* and *Overwatch* ends. If *specific Condition* was declared by another *Character* an *Interrupt* is triggered. The *Character* on *Overwatch* receives a +9 *Modifier* for the *Reflex Test* in this case.

Specific Conditions should not be too complex and should be easily identified as true by the *Character* on *Overwatch* with the current situation.

Specific Actions can only be *Active Actions*.

A *Character* can switch from *Overwatch* to *Delay* at any time.

Move Normal movement, like crawling, walking or running, can be performed in addition to *Active Actions*. However, if of another *Action* is performed while moving, a *Movement Modifier* is applied.

The *Movement Modifier*, as well as the distance moved depend both on the movement style as well as the duration of the movement, which is represent by the *Action Time*. The Character moves a number of meters according to their relevant *Movement Rate* and *Action Time*.

A *Move Action* can only performed in addition to an *Active Action* if the *Action* does not include movement.

Table 3.3: Movement Modifiers

Modifier	Movement Style
-6	Crawl
-1	Walk
-3	Run

Talk When a *Character* is talking wile performing a *Test*, the *Test* receives an additional -1 *Modifier*.

Table 3.4: Talk

Action	Words
Free	1 Word
Simple	Short sentence
Half	2 Short sentences
	1 Long sentence
Full	4 short sentences
	2 Long sentences

Ongoing Actions *Ongoing Actions* represent *Actions* that take, longer than 20 *Ticks*, sometimes considerably longer. They are represented by increasing the *Initiative Score* of the *Character* performing the *Action* by 20 if the *Character* wants to continue doing the *Ongoing Action*, till it is finished.

3.2.3 Continuous Inclusive Actions

Leadership

Tactics

3.2.4 Active Actions

Active Actions are *Actions* a *Character declares* when their *Initiative Score* matches the *Current Tick*. They are the default action type.

3.2.4.1 General Actions

Free Actions Free Actions without specific rules include:

- dropping an object
- change facing 90°

Simple Actions Simple Actions without specific rules include:

-
- change facing 180°

Half Actions Half Actions without specific rules include:

- change stance

Full Actions Full Actions without specific rules include:

- perform a *Perception Test*
- perform a *Jump Test*

Change Posture
Action Half
Test None/Acrobatics-6

Change posture from:

- standing to crouching
- crouching to prone
- standing to prone
- prone to crouching
- crouching to standing

Changing two levels at once (in this case only prone to standing) requires an *Acrobatics -6 Test*.

Drop Down
Action Free
Test None

Dropping down from a standing posture. After doing so, the *Character* is *disoriented*.

Ready Weapon
Action var.
Test None/Quickdraw

Ready a weapon given that the weapon is carried on the *Character*. A weapon needs to be ready to perform any *Actions* with it.

Table 3.5: Ready Weapon

Weapon	Action	Test
2 handed	Full	None
2 handed	Half	Quickdraw -8
2 handed	Simple	Quickdraw -16
1 handed	Half	None
1 handed	Simple	Quickdraw -8
1 handed	Free	Quickdraw -16

Reload Weapon
Action var.
Test None

Reloads a weapon given that ammunition is carried on the *Character*. Reload time depends on the weapon.

Sprint
Action Full
Test Running

Move number of meters equal to the *Characters Full Running Rate*. Add 1 meter for every 2 *Test Quality* of the *Running Test*.

Zigzag
Action Full
Test None
The *Character* moves a number of meters equal to half their *Running Rate*. All attackers get the "Running 90° to Attacker" *Modifier*. Using the *Zigzag Action* is not possible when carrying an *unwieldy* weapon and not exceeding its *Required Strength* by 3.

3.2.4.2 Ranged Actions

Ranged Actions are specific to ranged combat.

Aim
Action Half/Full
Test None
Damage No

The *Character* aims at a target in their *Front* or *Side Arc*. The gain a +1 *Modifier* for all *Ranged Actions* that do *Damage* on the aimed target. This Action can be taken multiple times till a maximum Modifier of +4 is reached. *Aiming* with a *Full Action* counts as *aiming* 3 times.
Equipment modifies aiming as follows:

- A Red Dot/Smartgun conveys a +2 *Modifier* for the first *Aim Action*
- A Rangefinder conveys a +2 *Modifier* for the second *Aim Action*
- A Scope allows for a maximum *Modifier* of +6
- A Gun Light reduces emphVisual Modifiers for the target if the target is in range

The *Aim Modifier* is lost when the target moves out of sight or the *Character* takes any other *Action* than:

- any Free Action
- any Ranged Action
- any Processing Test
- any Empathy Test
- Talk
- Walk

Brace Weapon
Action Full/Half
Test None
Damage No

A *Character* braces their two handed ranged weapon against a suitable object like a sandbag a window sill or a railing. The *Weapon Range* is doubled for all *Ranged Attacks* against targets in the *Characters Front Arc* with a height difference no greater than 1/4th of the distance.
The *Required Strength* of the weapon is also reduced by 3.
If the weapon is equipped with a bipod, bracing takes only a *Half Action* and the height difference can be 1/3rd of the distance.
Weapons equipped with a tripod take a *Full Action* to *brace*, which means they are effectively deployed. In this case *bracing* is not lost when the *Character* moves, if the *Character* choses to let go of their weapon. *Changing Facing* and *Posture* is also possible without losing the *braced* effect. The maximum height difference is 1/2nd.

Bracing for non tripod-weapons is lost when the *Character* moves in any way (also as a result of a *Dodge*, changes facing or changes posture).

Fast Ranged Attack

Action	Half
Test	Ranged Weapons -3
Damage	Yes

The *Character* performs a *Ranged Attack* with a -3 *Modifier*. Weapons with the *Single Shot* keyword can not perform this *Action*.

Burst Ranged Attack

Action	Full
Test	Ranged Weapons +2/+3
Damage	Yes

The *Character* performs a *Ranged Attack* with a weapon that has a *Short Burst* or *Long Burst* value. Short Bursts add a +2 *Modifier*, Long Bursts a +3 *Modifier*. For *Damage Resolution* multiply both *Armor* of the target and *Penetration* of the weapon with 2 (Short Burst) or 3 (Long Burst).

Burst *Ranged Attacks* can have an increased *Required Strength*.

Multi Ranged Attack

Action	Full
Test	Ranged Weapons -2/-1 per attack, -1 per additional target
Damage	Yes

The *Character* is performing multiple *Ranged Attacks* against one or more targets. Only one *Test* is rolled by the *Character*, *modified* by -2 per attack and -1 per additional target. This value is opposed by all targets individually.

This *Action* is only possible with one handed ranged weapons that do not have the *Single Shot* keyword. Wielding two ranged weapons decreases the attack *Modifier* to -1 and allow for 2 attacks even with *Single Shot* weapons.

Ranged Attack

Action	Full
Test	Ranged Weapons +0
Damage	Yes

The *Character* performs a standard *Ranged Attack*.

Suppression Fire

Action	Full
Test	No
Damage	No

The *Characters* suppresses an area with *Suppression Fire*. The attacker can cover a length of meters equal to the number of bullets of the weapons *Suppression Fire* value. Everybody moving through the suppressed zone or not going prone/into cover while being in it is hit with a final *Test Quality* of d10. Add +1 to the result if a target is crossing the area. The suppressed zone lasts till the *Characters Initiative Score*.

Suppression Fire can not be used if the*Required Strength* of the weapon is not met by the *Character*.

3.2.4.3 Melee Actions

Advance

Action	Full
Test	*Melee Weapons* +3
Damage	Optional

The target of this *Melee Attack* must retreat 1 meter for every two points of *Test Quality*, or be hit with the *Test Quality*. If the target choses to retreat, the *Character advances* the same number of meters.

Charge

Action	Full
Test	*Melee Weapons* -2
Damage	Yes

This *Action* requires the *Character* to run in a straight line to the target using their *Run Rate*. If the Melee Attack is successful the *Character* can add 1 point for every 2 meters crossed to the weapon damage.

Disarm

Action	Full
Test	*Melee Weapons* -3 -X
Damage	No

If this *Melee Attack* is successful, an additional opposed *Strength Test* is performed with a *Modifier* of X for the *Character*. If the *Character* wins this *Test*, the target is disarmed.

Disengage

Action	Full
Test	*Melee Weapons*
Damage	No

If this *Action* is successful, the *Character* can move a number of meters equal to their *Walking Rate* without risking a *Free Strike* from the target.

Feint

Action	Full
Test	*Melee Weapons* +3
Damage	No

If this *Action* is successful, the *Character* receives a +3 *Modifier* for the next *Melee Action* against the target.

Fast Melee Attack

Action	Half
Test	*Melee Weapons* -3
Damage	Yes

The *Character* is making a *Melee Attack*.

Melee Attack

Action	Full
Test	*Melee Weapons*
Damage	Yes

The *Character* is making a *Melee Attack*.

Multi Melee Attack

Action	Full
Test	*Melee Weapons* -2/-1 per attack, -1 per additional target
Damage	Yes

The *Character* is performing multiple *Melee Attacks* against one or more targets. Only one *Test* is rolled by the *Character*, *modified* by -2 per attack and -1 per additional target. This value is opposed by all targets individually.

Wielding two melee weapons decreases the attack *Modifier* to -1 for attacking a single target multiple times, while a double handed weapon decreases the attack *Modifier* to -1 for attacking multiple targets once (still adding -1 for each additional target).

Power Attack

Action	Full
Test	*Melee Weapons* -2
Damage	Yes

The target gets an additional negative *Modifier* of twice the *Strength* difference between the *Character* and the target.

Precise Strike

Action	Full
Test	*Melee Weapons* -2
Damage	Yes

If successful, this *Action* adds 2X to the *Melee Attack Test Quality*.

3.2.5 Reactions

In contrast to *Actions*, *Reactions* are *declared* as a reaction towards an *Action* of another *Character* or an event in the game world. No *Interrupt* is required to *declare* them. Most *Reactions* define precisely how they interact with the triggering *Action*.

Apart from this, *Reactions* are normal *Actions* with all rules and limitations, especially the *Maximum Initiative Score*.

3.2.5.1 General Reactions

Dodge

Action	Half/Simple/Free
Test	Dodge-2/-3/-4

The *Character* is dodging an incoming *Attack Action* that the *Character* is aware of, turning the attack into an *opposed Test*.

The *Character* can chose to drop to the ground while *dodging*, adding a +2 *Modifier* but also being *disoriented* afterwards.

If the *Character* is not dropping down, they may chose to move 1 meter for for every 3 points of *Test Quality*.

3.2.5.2 Melee Reactions

Disarm

Action	Full
Test	*Melee Weapons* -3 -X
Damage	No

The *Character* is opposing a *Melee Attack*. If this *Reaction* is successful, not only is the attack avoided, but an additional opposed *Strength Test* is performed with a *Modifier* of X for the *Character*. If the *Character* wins this *Test*, the target is disarmed.

Free Strike

Action	Half
Test	*Melee Weapons*
Damage	Yes

The *Character* performs a *Melee Attack* against a target that is moving out of their *Control Area*.

Masterful Parry

Action	Half
Test	*Melee Weapons* -1 -X
Damage	No

The *Character* is opposing a *Melee Attack*. If this *Action* is successful, the *Character* is getting a +X *Modifier* for their next *Action* against the attacker.

Parry

Action	Half/Simple/Free
Test	*Melee Weapons* 0/-1/-2
Damage	No

The *Character* is opposing a *Melee Attack*.

Riposte

Action	Full
Test	*Melee Weapons* -3
Damage	No

The *Character* is opposing a *Melee Attack*. If the *Riposte* is successful the attacker is themselves hit and needs to resolve a hit with the *Test Quality* of the *Riposte Action*.

3.3 Attack Resolution

3.3.1 Ranged Combat

3.3.2 Melee Combat

3.4 Hit Resolution

3.4.1 Hit Location

3.4.2 Damage

Chapter 4

Computers

This chapter explains both the matrix, including AR and everything computer related like electronic warfare.

Matrix rules are inclusive rules in a sense that each and every action a Player can take is described. This is in contrast to real world actions where the rule system gives a broad framework for players to extrapolate. This is the case because every Player has real world experience and expectations which just need to calibrated by rules (like introducing cyberware).

In the Matrix however, there is no common experience and thus no a priori contract between Players and Game Master what is possible and how probable it is. Thus Matrix rules have to be very strict and not assume anything.

As a rule of thumb, a character can not do anything that is not a given *Matrix Action*. However, the nature of the Matrix allows to describe the same *Action* very differently.

4.1 What is the Matrix

The Matrix is a virtual representation of the cyberspace for human users. It is they way they perceive interactions between themselves and both other matrix users and *Matrix Entities*.

4.1.1 Accessing the Matrix

There are various ways to access the Matrix.

Physical Access This method of *Matrix Access* uses outdated methods like keyboard and mouse. It is generally outdated and very slow. It is only used if people are afraid of any kind of matrix damage, or are very traditional.

Augmented Reality Augmented Reality or AR access is a widely used for of matrix access, especially one the go or while wanting to do things in parallel. AR users still see the real world, but get additional information projected on top of it. Thus they can see objects, additional information and also sound added to the real world that does not exist.

Virtual Reality Virtual Reality supersedes the perception of the user. They are not aware of the real world, but instead see, hear, smell and feal virtual sensory input that is 100% artificial.

Tortoise Tortoise uses not direct brain interfaces as provided by most data jacks, but uses outdated technologies like trodes. Due to it not requiring cyberware it is often used by adepts or magicians.

Cold Sim Cold Sim is the standard way of using the matrix today. The user is experiencing the matrix by direct stimulation of their sensory cortex so that they see, hear and feel the matrix. Their thoughts of movements and actions are translated into commands of their virtual bodies using virtual applications.

Hot Sim Hot Sim is the most dangerous but also the fastest way to access the matrix. The data is directly fed into the users brain even circumventing their sensory centers that are stimulated in cold sim. Instead, using knowledge link technology, the matrix user just instantly knows the information. Also their raw thoughts are transformed into matrix commands.

Table 4.1: Matrix Access Methods

Method	Input	Output
Physical	• Keyboard • Mouse • Touchscreen • Input Trigger	• Screen • Loudspeaker
AR	• Transducer • Microphone • AR Gloves • Holo Scanner	• Lenses • Vision-Link • In-Ears • Sound-Link
Tortoise	• Trodes • External Sim Rig	• Trodes • External Sim Module
Cold Sim	• Sim Rig	• Sim Module
Hot Sim	• Transcriber	• Knowledge Link

4.2 Matrix building blocks

4.2.1 Matrix Devices

The Matrix is made up of hardware that is processing and delivering it. Most notable are are the different pieces of hardware the matrix is running on. In general four different classes of matrix hardware can be found.

Table 4.2: Matrix Access Requirements

Method	Processor/ Uplink
Physical	1
AR	3
Tortoise	6
Cold Sim	6
Hot Sim	10

Table 4.3: Matrix Access Modifiers

Method	Skill	React	Tick	Damage
Physical	-3	-5	x6	None
AR	-2	-3	x3	Fatigue
Tortoise	-1	-2	x1.5	Fatigue
Cold Sim	0	0	x1	Stun
Hot Sim	+2	+3	x0.7	Physical

Gadget Gadgets are small and cheap pieces of hardware. Some of them are so cheap, they can be found in throwaway articles like food packaging. Others are powering small sensors or track positions. They range from pinhead size to coin size. A typical person is carrying around dozens of them.

Commlink Commlinks are not only the most common mans to communicate but also a matrix hardware class. They are bigger than gadgets, but the smallest of them can fit into a bigger earring. The standard size is of an average playing card. They carry enough processing power to allow for at least *Augmented Reality*.

Cyberdeck Cyberdecks are a special form factor that only few people need. Much bigger than a an average commlink, about the size of a shoe-box, they pack much more processing power. Most cyberdecks are used for illegal purposes and are equipped with a *Sleaze* module to avoid detection in the matrix.

Mainframe Mainframes are stationary pieces of matrix hardware. They range from shoe-box size to whole floors of a building. Mainframes are used to service multiple people or perform high performance computations.

4.2.2 Matrix Entities

Matrix entities are virtual building blocks of the matrix. Although they have a physical basis, they are purely virtual representations both in virtual- and augmented reality.

Node A Node is a matrix entity with processing power. It has matrix location and can be *accessed*. A Node can run *Processes*, store *Files* and be the origin or destination of a *Stream*.

Process Processes are matrix entities that actively perform actions. They are running on their origin *Node*.

Persona A Persona is a special kind of *Process* that represents a matrix user and their actions. *Personae* can *access Nodes*. In this case they are connected to their *origin Node* via a *Stream*.

Program A program is a piece of software that can be used by a *Persona* or an *Agent* as a tool to perform various actions. Programs are always attached to a *Persona* or *Agent*.

Agent An agent is a process that can perform autonomous decisions and use *Programs* to perform actions. *Agents* can *access Nodes*. In this case they are connected to their *origin Node* via a *Stream*.

ICE ICE, or Intrusion Countermeasures, are *Agents* with the special purpose to defend a node from hackers.

Streams A stream connects two *Nodes*, the origin and the destination, with a data connection. A stream also connects the *Node* a *Persona* or *Agent* is running on with the *Node* it is *accessing*.

File A *File* is a coherent set of any kind of data.
This includes:

- a text document
- a trideo clip
- a BTL movie
- a voice record

4.2.3 Access Levels

In Pink Trenchcoat a decker that is *accessing* a *Node* is identified with a given *Access Level*, or *Account*. This *Account* is specific to the *Node* and linked to the deckers SIN or, in the case of *Agents*, to their *AID*.

Anonymous *Anonymous Access Level* is the default *Level* that is automatically granted to everyone.

User *User* is a catch all *Level* for a large number of *Accounts* of different Matrix users. A *Node* can have multiple *User Levels* with non overlapping *Access Rights* for *Files*, *Streams* and *Processes*. If a decker *Exploited* a *User Account* the Game Master decides which *Access Rights* come with it.

Security The *Security Level* is, in addition to any or no *User Access Rights* used to perform various security relevant *Actions*, especially controlling *ICE* and maintaining the *Security Tally*.

Admin The *Admin Access Level* can do everything in a *Node*.

4.2.4 Matrix Properties

Access Rights Each *Matrix Entity* has *Access Rights* that govern which *Account Levels* are allowed to perform which *Actions*. These *Rights* govern for example who can *Access* a *Node*, *Read* a *File*, *Send* to a *Stream* or *Command* a *Process*.

Each *Node* their own *Access Rights* for the *Matrix Entities* they contain. They can be changed by the *Edit Access Rights Action*.

Access ID

Subscription List

Logs The *Logs* are a special *File* that contains a history of all actions in a Node, including all actions of *Personae* and *Agents*, their *AIDs*, the *Files* and *Streams* the created and consumed and anything else that was done in the *Node*. *Actions* from a *Process* that has a *Sleaze* rating are only *logged* when they have been successfully *analyzed* by *Analyze ICE*.

Integrity Each *Matrix Entity* has an Integrity value that is a measure for how much *Matrix Damage* it can take before it is suffering negative consequences.

The Integrity value is depending on the origin *Node* of the *Entity*:

$$Integrity = 10 \cdot System \tag{4.1}$$

Matrix Damage *Matrix Damage* done to an *Entity* is added up until it reaches *Integrity*. Once the *Damage* reaches this threshold *Nodes* are *shut down*, *Processes* are *crashed*, *Files* are *deleted* and *Streams* are *terminated*.

4.2.5 Matrix Attributes

Each *Matrix Device* has a number of attributes that define its properties in the Matrix.

Processor The *Processor* attribute represents a *Nodes* row computing power. As most devices are very advanced, a high *Processor* rating is not needed for most every day tasks. High *Processor* ratings are required for intensive tasks like processing Sim-Sense signals for example when using *Cold Sim* or the even more complex *Hot Sim*. The attribute is also useful if a mainframe is supporting a large user base.

It is also important in matrix combat where combatants try to overwhelm the opponents *Node*.

The *Processor* attribute is mostly related to a *Devices* size. The bigger a *Device* the higher its rating is on average.

Table 4.4: Processor Ratings

Entity	Processor
Gadget	0–4
Commlink	3–8
Cyberdeck	6–13
Mainframe	8–21

System System describes the quality of the operating system and standard software suite of a *Node*. The higher the ranking the higher the rating of *Programs* that can be run.

A high Systems rating also helps autonomous software like *ICE* to perform more efficiently.

Firewall Firewall represents the resilience of a *Node* against anything illegal. This includes any kind of *Exploit* actions leading to illegal actions not governed by the users level.

Firewall is not determined by a *Nodes* computing power but by the skill and time invested by the maintainers of the node, and the number of users and different *Processes* it is supporting.

Firewall Ratings are often given by a color coding.

Table 4.5: Firewall Ratings

Color	Firewall
Blue	0–4
Green	5–9
Orange	10–14
Red	15–19
Ultra Violet	20–21

Blue Blue *Nodes* represent the lowest level of security. They are often either very cheap gadgets like Smart Tags or public mainframes like public libraries.

Green Green *Nodes* represent the vast majority of matrix hosts. They are a good trade-off between expensive security experts and time invest. *Nodes* with fewer users tend to have higher green ratings.

Orange Orange *Nodes* are used when higher security is required, like in the mainframe of a police station, a law firm, or the *Nodes* of upper class individuals.

Red Red *Nodes* are mostly used by high security facilities like corporate research sites or government agencies.

Ultra Violet Ultra Violet *Nodes*, if they exist, are only used for legendary and top-secret institutions.

Uplink Uplink describes the quality, speed and volume of data that a *Node* can access per time. A high throughput is required for *Cold Sim* and even more for *Hot Sim*. Uplink mostly degrades over distance, although not as fast as wireless *Signal* does, or if the signal has to go through wireless channels.

Signal The Signal rating describes the power and quality of a wireless signal. It is used to check how far a signal penetrates and also represents the power delivered in case

of *Electronic Warfare*. Only nodes with wireless capabilities have a Signal rating.

Table 4.6: Signal Ranges

Signal	Range	Signal	Range
0	1 m	**11**	5 km
1	2 m	**12**	10 km
2	5 m	**13**	20 km
3	10 m	**14**	50 km
4	20 m	**15**	100 km
5	50 m	**16**	200 km
6	100 m	**17**	500 km
7	200 m	**18**	1,000 km
8	500 m	**19**	2,000 km
9	1 km	**20**	5,000 km
10	2 km	**21**	10,000 km

Sleaze Only devices equipped with with an illegal *Sleaze* module have a *Sleaze* rating. The *Sleaze* rating allows a decker to hide from security software of a *Node*. Without it the decker would instantly be recognized after performing any kind of *Exploit* action.

A *Sleaze* module allows also to broadcast and change (fake) SINs the decker possesses. The decker can not mimic arbitrary SINs.

4.3 Matrix Concepts

4.3.1 Security Tally

The *Security Tally* is tally that is specific for each *Process accessing* a *Node*. It is a measure on how suspicious a *Node* is about illegal *Actions* from a *Process*. The *Tally* increases by performing illegal *Actions* while not having a high enough *Sleaze Attribute* to not be noticed.

Actions that increase the *Security Tally* include:

- Exploit
- Crash
- Corrupt

The *Tally* for a *Process* can be changed by using the *Edit Logs Action*. Various effects of the Tally like *ICE* and *Alerts* can also be reverted by various *Security Actions*.

Depending on the value of the *Tally*, the *Node* is launching various countermeasures.

Table 4.7: Security Tally Measures

Tally	Measure
5	Analyze ICE
10	Trace ICE
15	Silent Alert
20	Combat ICE
25	Active Alert
50	Emergency Shutdown

Analyze ICE Analyze ICE is looking into a deckers activities to find any signs of illegal actions. If it finds anything it will be added to the deckers security tally.

While Analyze ICE is not *slowed*, it is adding 2 points to the *Nodes System* for any interaction, active or passive, with the triggering *Process*. This includes calculation of *Security Tally* increase as well as *Analyze Action* or *Crash Actions*.

If it is *crashed* it takes 10 seconds to restart.

Trace ICE Trace ICE will try to find the deckers location by analyzing its *Stream*. It will immediately start to perform an *Analyze Action* against the the triggering *Process*. Any information gained, especially location, is written in the *Nodes Logs*. The Nodes *System* is used as the *Ability Score* for the *Analyze Test*.

If it is *crashed* it takes 10 seconds to restart and will need to restart the *Analyze Action*.

Passive Alert In *Silent* or *Passive Alert Status* a list of predefined personnel is informed of a possible intrusion. The *Node* diverts resources to security purposes, increasing *Firewall* by 2 and decreasing *Processor* by 2. Any standard functionality of the *Node* could be impaired by this resource transfer (GM discretion). The information is not broadcasted to *Processes* in the *Node*.

Combat ICE Combat ICE will once triggered continuously attack the triggering *Process* in the form of *Crash Actions*. The Nodes *System* is used as the *Ability Score* for the *Crash Test*.

If it is *crashed* it takes 10 seconds to restart.

Active Alert In active Alert Status a list of predefined personnel is informed of an intrusion. The *Node* diverts resources to security purposes, increasing *Firewall* by 3 and decreasing *Processor* by 3. This is not cumulative with the changes made by the *Passive Alert Status*. Any standard functionality of the *Node* can be impaired by this resource transfer (GM discretion). The information is broadcasted to all *Processes* in the *Node*.

Emergency Shutdown

4.4 Matrix Actions

4.4.1 Basic Actions

Basic Actions are very simple and normally do not require a *Test* or *Program*. If a *Test* is required because the *Character* is wounded or has an extreme non-technical background use:

$$Computers + 3$$

Access Node
Prerequisite *Node AID*
Duration 0.1s

Table 4.8: Matrix Actions

Account Level	Program	Node	Process	Stream	File
Anonymous	None	• Anonymous Access			
	Analyze	• Analyze	• Analyze	• Analyze	• Analyze
	Break			• Break	• Break
	Corrupt	• Crash • Slow	• Crash • Slow	• Corrupt	• Corrupt
	Find	• Find	• Find	• Find	• Find
User	None	• User Access	• Command • Start • Stop	• Decrypt • Read • Start • Send • Terminate	• Create • Decrypt • Delete • Read • Write
	Control		• Control [Thing]		
	Crypt			• Encrypt	• Encrypt
	Generate			• Generate	• Generate
	Medic	• Repair	• Repair		
Security	None	• Security Access • View Accounts • View Alert Status • View Logs • View Subscriptions	• Command ICE • Start ICE • Stop ICE		
Admin	None	• Admin Access • Change Alert Status • Edit Access Rights • Edit Logs • Edit Subscriptions • Shutdown			

This action is required to access a *Node* with a known *AID*. After a successful *Access Action* the decker has *accessed* the *Node*.

Having *accessed* a *Node* is often a prerequisite for lots of *Matrix Actions* targeting *Files* and *Streams*. It is only of particular relevance when a decker does not have the relevant *Access Rights* to *access* the *Node* and needs to *Exploit* their way in.

Change Alert Status

Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to change the *Nodes Alert Status*.

Command

Prerequisite *Process AID*, *Accessed origin Node*
Duration 2s

This action allows a decker to give commands to a *Process*. This can either be an *Agent*, or any other *Program* on a *Node* or *Device* like a drone or a security camera.

The decker needs the *AID* of the *Process* and needs to *access* the origin or target *Node* of the *Process*.

Create File

Prerequisite *Accessed Node*
Duration 1s

This action creates a *File* in a *Node*. The creator chooses content and *Access Rights* and gets the *Files AID*.

Decrypt

Prerequisite *Red File*, *CryptKey*
Duration 0.1s

Decrypt and *encrypted File* if the decker has the *CryptKey*.

Delete File

Prerequisite *File AID*, *Accessed Node*
Duration 0.1s

Delete a *File* in a *Node*. After the *File* is *deleted* it can not be recovered.

Edit Access Rights

Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to edit *Access Rights* of a *Nodes*. This includes removing, adding and changing

Access Rights. In the case of adding a new *Account* the respective SIN is required.

Edit Logs
Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to edit the *Logs* of a *Node*. This includes adding and removing entries.

Edit Subscriptions
Prerequisite *Accessed Node*
Admin Access other Node
Duration 0.5s

This action allows the decker to edit the *Subscription List* of a *Nodes*. This includes removing and adding *Nodes*. In the case of adding a decker needs Admin Access on the other *Node*.

Read File
Prerequisite *File AID*, *Accessed Node*
Duration 0.1s

This action allows a decker to read *Files* in a *Node*. *Reading* a *File* enables a decker to *create* a local *File* copy in the *Personas* origin *Node*.

Read Stream
Prerequisite *Stream AID*
Accessed origin/target Node
Duration 0.1s

This action allows a decker to read *Streams* in a *Node*. *Reading* a *Stream* enables a decker to *create* a local *File* containing the content of the *Stream* in the *Personas* origin *Node*.

Start Process
Prerequisite *Accessed Node*
Duration 1s

This action creates a *Process* in a *Node*. The creator chooses its *Access Rights* and gets the *Process AID*.

Send to Stream
Prerequisite *Stream AID*
Accessed origin Node
Duration 1s

This action creates a *Stream* between two *Nodes*. The creator chooses content and *Access Rights*.

Start Stream
Prerequisite *Accessed origin Node*
Accessed destination Node
Duration 1s

This action creates a *Stream* between two *Nodes*. The creator chooses content and *Access Rights* and gets the *Streams AID*.

Stop Process
Prerequisite *Process AID*
Accessed origin Node
Duration 1s

This action *stops* a *Process*. A related *Agent* or *Persona* is instantly shut down.

Terminate Stream
Prerequisite *Stream AID*
Accessed origin Node
Duration 0.5s

This action *terminates* a *Stream*. A related *Process* is instantly stopped.

View Accounts
Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to view all *User*, *Security* and *Admin Accounts* for the *Node*.

View Alert Status
Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to view the current *Alert Status* of the *Node*.

View Logs
Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to view the current *Logs* of the *Node*.

View Subscriptions
Prerequisite *Accessed Node*
Duration 0.5s

This action allows the decker to view the *AIDs* of the *Nodes* the are *subscribed* to the *Node*.

Write to File
Prerequisite Found *File*
Duration 0.5s

This action allows a decker to *write* any content to a *File*.

4.4.2 Advanced Actions

Advanced Actions require *Tests* to perform and require a *Program* to carry out. The standard format of a *Test* is given as:

$$Skill(Program) + Test\ Modifier$$

Analyze [Node, Process, Stream, File]

Program	Computer(Analyze)
Prerequisite	Found [Node, Process, Stream, File]
Test Modifier	-Target *Sleaze*
Duration	2s

This action allows for analyzing properties of various matrix entities. To analyze a *Node* an AID is required. Other entities have to be *found*. *Processes* and *Streams* can only be analyzed if the decker has *accessed* either the target or the destination *Node*.

Table 4.9: Analyze Node Results

Result	Properties	Location
0	Active Alert Status	
2	AID	
4	Type	
6	High/Low Attributes	Continent
8	Functionality	State
10	High/Med/Low Attributes	City
12	Active Processes	Suburb
14	Exact Attributes	Street
16		Building
18		Room
20		Exact

Control

Program	Skill(Control)
Prerequisite	*Accessed Node* *Process AID*
Test Modifier	var.
Duration	1s

Using the *Control Action* the decker can use any kind of item that can be *controlled* remotely from a *Process* in a *Node*. The decker has to use the relevant *Skill* limited by the *Control Program*.

Skill(Control)

Examples are using remotely controlled guns using *Gunnery* or driving a remotely controlled car using *Wheeled*.

Encrypt

Program	Crypt
Prerequisite	*Accessed Node* *File AID* or *Stream AID*
Test Modifier	0
Duration	1s

The *Encrypt Action* encrypts a *File* or *Stream* so that even deckers with the required *Access Rights* can not read the content. The *Action* does not require a *Test* but automatically encrypts the *File* or *Stream* with the *Crypt Programs* rating. To read the content one needs either the key or try to *Break* the encryption.

Find Process

Program	Computer(Find)
Prerequisite	*Access* to origin/destination *Node*
Test Modifier	-Target *Sleaze*
Duration	10s

This action allows a decker to find *Processes* in a *Node*, which must be either its origin or the destination.

Find Stream

Program	Computer(Find)
Prerequisite	*Access* to origin/destination *Node*
Test Modifier	-Target *Sleaze*
Duration	10s

This action allows a decker to find *Streams* in a *Node*, which must be either its origin or the destination.

Find File

Program	Computer(Find)
Prerequisite	*Access* to origin/destination *Node*
Test Modifier	-Target *Sleaze*
Duration	10s

This action allows a decker to find *Files* in a *Node*.

4.4.3 Matrix Combat

Actions in this section directly harm *Matrix Entities* either by damaging or slowing them. Most of the time, *Processor* is an important *Attribute* in *Matrix Combat*.

Corrupt

Program	Cyber Combat(Corrupt)
Prerequisite	*Accessed Node* *File AID* or *Stream AID* or Found [*File*/*Stream*]
Test Modifier	-Originating *Node* System
Duration	1s

If a decker does not have the *Access Right* to *delete* a *File* or *Terminate* a *Stream* the decker can *corrupt* it so it becomes unusable.

For each point of *Test Quality* deal *Processor Matrix Damage* to the *File* or *Stream*.

In addition each *Corrupt Test* that can increase the deckers *Security Tally*. The increase is given by:

$$Tally = \max(0, 5fN + System - Sleaze)$$

Crash

Program	Cyber Combat(Crash)
Prerequisite	Found [*Node*/*Process*]
Test Modifier	-System
Duration	1s

For each point of *Test Quality* deal *Processor Matrix Damage* to the *Node* or *Process*.

In addition each *Crash Test* that is targeting a *Node* can increase the deckers *Security Tally*. The increase is given by:

$$Tally = \max(0, 2 * 5fN + System - Sleaze)$$

Repair

Program	Computer(Medic)
Prerequisite	AID
Test Modifier	0
Duration	10s

The *Repair Action* allows a decker to *repair Matrix Damage* on *Nodes*, *Processes*, *Files* and *Streams*. For each point of *Test Quality* repairs one point of *Matrix Damage* to the target.

Slow Node

Program	Cyber Combat(Slow)
Prerequisite	*Access* to *Node*
Test Modifier	-System
Duration	1s

The *Slow Action* allows a decker to reduce the *Processor* a target *Node*. Reduce the *Processor* by the *Test Quality*. This effect lasts for 10s, or till the decker takes another *Action*, whichever happens later.

Slow Process

Program	Cyber Combat(Slow)
Prerequisite	Found *Process* or AID
Test Modifier	-System
Duration	1s

The *Slow Action* allows a decker to increase the *Tick Cost* of *Actions* a target *Process*, including *Personae* and *Agents*. Increase Free, Simple, Half, Full Actions by 1,2,3,4 times the *Test Quality*. Longer *Action* get increased by 50% for each point of *TQ*. This effect lasts till the targets next *Action*, or till the decker takes another *Action*, whichever happens later.

4.4.4 Hacking

Break

Program	Hacking(Break)
Prerequisite	*Read File*
Test Modifier	-Crypt Rating -3
Duration	$10 \cdot \max(1, \textit{5fN})$ seconds

A successful *Break Test* removes the *Encryption* from a *File*. The *Duration* is determined by the *5fN* roll of the *Game Master* to a minimum of 10 seconds.

Exploit Every time a decker wants to perform an *Action* where their *Access Level* is not high enough, like *Viewing* the security *Log* without being at least *Security* level, an *Exploit Test* is required. If the *Action* in question requires a *Test* itself, when for example *Writing to a Stream*, the *Exploit Test* does not replace the actual *Test* but is an additional requirement.

An Exploit test is an opposed test between the deckers *Hacking(Exploit)* and the *Nodes Firewall*.

$$\textit{Test Quality} = \textit{10fR} + \textit{Hacking(Exploit)} - \textit{Firewall}$$

In addition each *Exploit Test* can increase the deckers *Security Tally*. The increase is given by:

$$\textit{Tally} = \max(0, \textit{5fN} + \textit{System} - \textit{Sleaze})$$

Table 4.10: Exploit Modifiers

Account Level	Mods	
	Action	Account
User	0	-3
Security	-3	-5
Admin	-4	-6

4.4.5 Related Actions

Physical Reboot

A *Physical Reboot* can only be done while having physical access to the *Node*. It does not require a test and takes time depending on the *Processor* of the *Node*:

$$\textit{Reboot Time} = \textit{Processor}^2 + \textit{seconds}$$

During a *Physical Reboot* the Admin Account can be changed to whatever *SIN(s)* or *AID(s)* the person that does the *Reboot* desires.

While the *Node* is rebooting all *Processes* are *Stopped*, all *Streams* terminated, and all *Subscriptions* are inactive. The *Node* can not be accessed and all *Processes Accessing* the *Node* are disconnected, maybe resulting in *Dump Shock*.

After the *Reboot* all *Matrix Damage* to *Processes*, *Files* and *Streams* is removed.

Data Search

Jack Out The *Jack Out Action* is required to prevent *Dump Shock*. As most *Matrix Access Methods* are at least a little and sometimes extremely immersive and taxing on the brain, and are thus requiring this Action to safely log off. The *Action* itself is a simple *Mental Composure Test* that takes 5 seconds independent of *Matrix Access Method*.

Disconnecting from the Matrix without *Jacking Out* results in *Dump Shock* which can happen in an a number of situations like an *Emergency Shutdown*

Dump Shock The decker takes:

$$\textit{damage} = 10 \cdot \textit{Mental Pip} \tag{4.2}$$

4.5 Electronic Warfare

Find Wireless

Program	Scan
Prerequisite	Target in *Signal* range
Test Modifier	var.
Duration	10s

Jam Wireless

Program	Scan
Prerequisite	None
Test Modifier	0
Duration	None

Chapter 5

Magic

5.1 Astral Space

5.1.1 Assensing

5.2 Invocation

5.2.1 Counterspelling

5.2.2 Spellcasting

5.2.3 Spontaneous Modifications

5.2.4 Ritual Spellcasting

5.3 Evocation

5.3.1 Banishing

5.3.2 Binding

5.3.3 Summoning

5.3.4 Spirits

5.3.5 Spirit Powers

Merialisation

5.4 Enchantment

5.4.1 Alchemy

5.4.2 Disenchanting

5.4.3 Enchanting

5.5 Meta Magic

5.5.1 Basic Techniques

Astral Projection

Centering

Divination

Masking

Shielding

Inner Binding

Flexible Signature

Inner Power

Infusion

Transfusion

5.5.2 Advanced Techniques

Manifestation

Extended Masking

Flux

Ally Formula

Quickening

Absorption

Reflection

Search

Shrouding

Spell Moulding

5.6 Adept Powers

Chapter 6

Spells and Powers

6.1 Spell List

6.1.1 Creation

6.1.1.1 Astral

Astral Barrier

Category	Astral Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create a Ward with a Rating equal to the *Test Quality*.

Astral Armor

Category	Astral, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create Astral Armor. Astral armor only exists on the astral plane.

6.1.1.2 Mana

Healing

Category	Mana, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Heal Physical Damage.

First Aid

Category	Mana, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Heal Physical Damage.

Heal Wounds

Category	Mana, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Heal Physical Wounds.

Increase Physical Attribute

Category	Mana, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Increase a physical attribute (Agility, Body, Coordination, Strength) by the [Task Quality]

Increase Rreflexes

Category	Mana, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Increase the Physical Reaction of the target by [Task Quality].

Mana Barrier

Category	Mana, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create a Barrier that only effects mana targets.

6.1.1.3 Physical

Barrier

Category	Physical, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create a physical barrier

Armor

Category	Physical, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create a magical armor around the caster.

Animate

Category	Physical, Creation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Animate a physical object.

6.1.2 Detection

Spells can only detect entities that have a presence on the plane the spell was cast. This means in particular that Mana spells cast in astral space an only detect astral and dual-natured entities.

6.1.2.1 Astral

Astral Window

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The Anchor is able to place his astral sense anywhere in the volume (thus removing his original astral perception). The Assensing of any Tests performed through the Spell are given by the Task Quality of the pellcasting Test. Spells can not be cast through the Astral Window spell. The Target has to use Astral Perception to benefit from the spell, as the spell is only shifting the targets original sense and does not grant Astral Perception in itself.

Detect Magic

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Detect any magic that is not disguised or disguised with a Task Quality less than the Task Quality of the Spellcasting test.

Detect Life

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Detect position and size of life depending on the the Task Quality of the Spellcasting test. A Task Quality of 0 enables the caster to pinpoint the position to within Volume/2 that is +/- Volume/4 meters and size to 2m (X +/-1m).

Detect Enemies

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Detect position of enemies depending on the the Task Quality of the Spellcasting. An enemy is defined as a sentient being that would, upon seeing the Anchor of the spell in the current location, want to take actions to decisively

harm the social, economical or physical well being of the Anchor. A Task Quality of 0 enables the caster to pinpoint the position to within Volume/2 that is +/- Volume/4 meters.

Detect Emotions

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Detect emotions depending on the the Task

Detect Individual

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Detect position and size of a sapient entity that was

Detect Thoughts

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Effect: Detect position and size of a sapient entity with given thoughts depending on the the Task Quality

Mindprobe

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Effect: See through the mind and thoughts of the target. The information gained depends on the the

Mindnet

Category	Astral, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Allow a mental connection between the target and a number of other sentient beings.

6.1.2.2 Physical

Catalog

Category	Physical, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Effect: Get a list and information about every of every object in the Volume. The detail of the information depends on the Task Quality of the Spellcasting test. The caster can not get more information than he would get if he would be able to see/inspect the object himself.

Clairaudience

Category	Physical, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The Anchor is able to place his aural sense anywhere in the volume (thus removing his original aural perception). The Visual Perception Skill Rating

Clairvoyance

Category	Physical, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The Anchor is able to place his visual sense anywhere in the volume (thus removing his original visual perception). The Visual Perception Skill Rating

Detect Sensors

Category	Physical, Detection
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Detect position and size of sensors on the the Task Quality of

6.1.3 Manipulation

6.1.3.1 Astral

Control Thoughts

Category	Astral, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The caster can control what the target wants to do. Doing so is a Complex Action. The Target is allowed to perform a Willpower test with a Difficulty of the Spellcasting test every time it receives an order that it is against its moral code (+0), endangers its mental/physical/economical well-being (+10) or life (+20) to break the spell.

Mana Illusion

Category	Astral, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Manipulate what everyone in the affected area perceives (Audio, Video, Olfactorial, Gustatory).

Fear

Category	Astral, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The target is instilled with fear of the caster. All Actions the target performs against the caster or anybody belonging to him are affected by a Task Modifier equal to the

Control Thoughts

Category	Astral, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The caster transforms the Target into another mundane creature. For every 10 Task Quality of the Spellcasting test the mundane creature can weight 1/2 or 2x the mass of the target.

6.1.3.2 Mana

Possession

Category	Mana, Body, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The caster takes control of the targets body as if it was his own (the casters original body behaves like it was in astral space). The caster can use all his mental skills (skills that have no physical mod contribution) and all the targets physical skills with a -10 modifier (in addition to any other modifiers like sustaining the spell). The Target is allowed to perform a Willpower test with a Difficulty of the Spellcasting test every time its body receives an order that it is against its moral code (+0), endangers its mental/physical/economical well-being (+10) or life (+20) to break the spell.

6.1.3.3 Physical

Physical Illusion

Category	Physical, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Creates a multisensor illusion (Audio, Video) that can only be detected with a perception test with a base Difficulty equal to the Task Quality of the Spellcasting test, or if somebody touches the illusion. A physical illusion spell is not able to simulate objects much hotter than Task Quality degrees Celsius above/below the current room temperature, as the spell has to effectively create/destroy radiation to achieve this effect. Also it can not generate sounds louder than 60dBA + Task QualitydBA as the spell has to effectively create sound waves. A physical illusion spell can not destroy light and sound coming from inside the illusion radius, only overlay it. It can thus not simulate the effects of silence or invisibility.

Physical Mask

Category	Physical, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Creates a multisensor volumetric illusion (Audio, Video) that can only be detected with a perception test with a base Difficulty equal to the Task Quality of the Spellcasting test. The physical mask spell allows the target to appear as another being of roughly the same size (+/- Task Quality percent). A physical mask spell can not destroy light and sound coming from inside the illusion radius, only overlay it. It can thus not simulate the effects of silence or invisibility.

Physical Invisibility

Category	Physical, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The spell manipulates photons in such a way that the target can almost not be detected visually. To see the flaws in the illusion a visual perception test with a base Difficulty equal to the Task Quality of the Spellcasting test needs to be passed.

Physical Silence

Category	Physical, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

The spell manipulates sound waves in such a way that the target can almost not be detected aurally. To hear the flaws in the illusion an aural perception test with a base Difficulty equal to the Task Quality of the Spellcasting test needs to be passed.

Levitation

Category	Physical, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Levitates a person or object.

Magic Fingers

Category	Physical, Manipulation
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Allows the caster to perform actions as if he were present at any location in the spell Volume. The fingers have an Agility.

6.1.4 Destruction

6.1.4.1 Astral

Astral Weapon

Category	Astral, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create Astral Weapon

Stunbolt

Category	Mana, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Direct Stun Damage

6.1.4.2 Mana

Manabolt

Category	Mana, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Direct Physical Damage

Sterilize

Category	Mana, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Reduction of Lethality of all diseases in the Volume by Task Quality/2 of the Spellcasting to connect biological samples to its owner (be it physical identification or magical tracking).

6.1.4.3 Physical

Firebot

Category	Physical, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create a Firebolt that start at the Range and flies toward a given target. This attack handles exactly like a ranged attack with a Skill equal to the characters Skill test.

Fireball

Category	Physical, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Create a Fireball that start at the RangePoint in Space and flies toward a given target. This attack handles exactly like a Area Attack with a Skill equal to the characters Skill The Scatter Length is Volume/10.

Wreck Vehicle

Category	Physical, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Direct Physical Damage

Melt Structure

Category	Physical, Destruction
Modifier	0
Time	60s
Drain	4 + 5fR
Range	Self
Volume	4m
Anchor	Self
Duration	Sustain

Direct Physical Damage

6.2 Critter Powers

Accident

Binding

Concealment

Confusion

Engulf

Ethereal

Fear

Fluid

Guard

Immunity to X

Increased Armor

Influence

Movement through X

Protection from X

Resistance to X

Sensitivity to X

Tolerance of X

Vulnerability to X

Structureless

Woundless The awakened does not suffer wounds.

Chapter 7

Resources

7.1 Lifestyle

7.2 Security

7.2.1 Physical

7.2.2 Matrix

7.2.3 Magic

7.3 Contacts

7.3.1 Organizations

7.3.2 Matrix

7.3.3 Magic

7.3.4 Media

7.3.5 Organized Crime

7.3.6 Law

7.3.7 Scene

List of Tables

List of Figures

Alphabetical Index

Appendix A

Game Tables

Table A.1: Ranges

Range	Range
1 cm	50 m
2 cm	100 m
5 cm	200 m
10 cm	500 m
20 cm	1 km
50 cm	2 km
1 m	5 km
2 m	10 km
5 m	20 km
10 m	50 km
20 m	100 km