VisoCrypt

Philipp Fleiß Mario Pivk Nina Winkler

16. April 2009

Inhaltsverzeichnis

Zusammenfassung

Der Sinn dieser Arbeit ist die Vorstellung des Verfahrens der visuellen Kryptographie. Wir beschreiben die Entwicklung eines Programmes zur Veranschaulichung des Verfahrens und dokumentieren die fertige Software und geben Vorschläge für neue Features.

1 Einleitung

Durch den Einsatz von Computern haben sich die Möglichkeiten für den Einsatz von kryptographische Verfahren stark erhöht, jedoch sind auch die Anforderungen mehr geworden. Sehr viele der klassischen Verschlüsselungsverfahren sind durch die Möglichkeiten der Kryptoanalyse mit Computern nicht mehr sicher zu verwenden. Man ist bei den meisten modernen kryptographischen Verfahren auf Computer angewiesen, was manchmal, wenn man keinen Computer zur Verfügung hat, zu Problemen führen kann. Da visuelle Kryptographie keine technischen Hilfsmittel benötigt und vollkommen sicher ist, könnte sie in gewissen Bereichen eine weite Verbreitung erfahren.

2 Visuelle Kryptographie

2.1 Einleitung

Erstmals wurde das Verfahren der Visuellen Kryptographie 1994 von Adi Shamir und Moni Naor in Eurocrypt94 vorgestellt. Das Grundprinzip ist folgendermaßen: Die Informationen eines geheimen schwarz-weißen Bildes werden so auf mehrere Folien verteilt, so dass die einzelnen Folien wie zufällige schwarz-weiß Muster wirken. Bei Übereinanderlegen der Folien jedoch wird das ursprüngliche geheime Dokument sichtbar. Im Gegensatz zu anderen kryptographischen Methoden benötigt man für die Entschlüsselung somit keinerlei kryptographische Kenntnisse oder komplexe Berechnungen. Die Entschlüsselung des geheimen Bildes erfolgt allein mit Hilfe des menschlichen Sehsystems. Da das Verfahren mit einem One-Time-Pad vergleichbar ist, ist es informationstheoretisch sicher.

2.2 2 aus 2

Bei dem 2 aus 2 - Verfahren, dem Originalproblem von Naor und Shamir, gilt es das geheime schwarz-weiße Bild derart auf zwei Folien zu verteilen, dass einerseits bei Übereinanderlegen der beiden Folien das geheime Bild sichtbar wird, aber auch dass andererseits jede Folie für sich alleine keinerlei Informationen bezüglich des geheimen Bildes preisgibt.

Wie man an Abbildung 1 sehen kann, ist es in keiner Weise möglich von der Information einer Folie zu schließen, ob es sich im Originalbild, um einen weißen oder schwarzen Pixel handelt. Des Weiteren wird jeder weiße Pixel nach Aufspaltung und Übereinanderlagerung zu einem weißen und einem schwarzen Subpixel während ein schwarzer Pixel zu zwei schwarzen Subpixel wird. Dies ist die Folge davon, dass unser Sehsystem OR-Verknüpfungen durchführt und nicht XOR - Verknüpfungen. Dadurch ergibt sich bedauerlicherweise ein Kontrastverlust, wodurch letztlich Verschlüsseln auf mehr als vier Folien keinen Zweck mehr hat, da bei Entschlüsselung, also bei Übereinanderlagerung der Folien, das Originalbild kaum beziehungsweise nur mit viel Eingebung erkennbar ist. Damit beim 2 aus 2 - Schema aufgrund dessen, dass wir hier nur zwei Subpixel haben, das Bild nicht gezerrt werden muss beziehungsweise damit man nicht skalieren muss, was ja einen weiteren Kontrastverlust zur Folge hätte, spalten wir jeden Pixel in vier statt zwei Subpixel pro Folie auf.

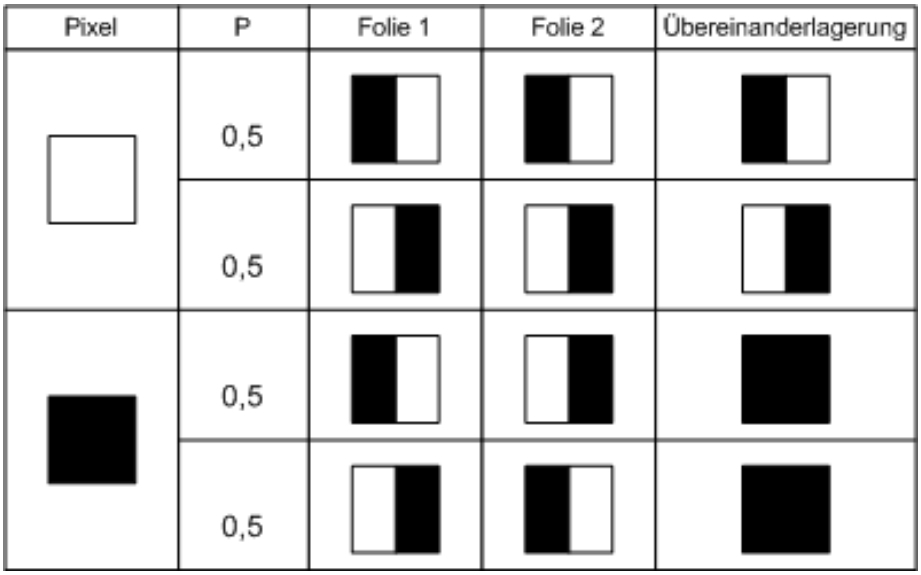

Abbildung 1: Verfahren: Jeder Pixel des geheimen Bildes wird in 2 Subpixel pro Folie aufgespaltet.

2.3 Modell

Die geschickteste Methode an das Visual-Sharing Problem heranzugehen, ist jeden Pixel des schwarz-weißen Originalbildes separat zu betrachten. Jeder Pixel wird auf n Folien verteilt verschlüsselt, wobei auf jeder Folie der n Folien der Pixel aus einer Menge von m schwarzen und weißen Subpixel besteht. Die Subpixel eines Pixels befinden sich dabei so nah beieinander, dass das menschliche Sehsystem die Subpixel als zusammengehörend wahrnimmt. Die resultierende Struktur wird durch eine boolsche n x m - Matrix S beschrieben, wobei s_{ij} genau dann den Wert 1 enthält, wenn in der i-ten Folie der j-te Subpixel schwarz ist. Ebenso bedeutet an der Stelle der Wert 0, dass der Subpixel weiß ist. Bei Übereinanderlegung der Folien, sieht man eine kombinierte Folie, bei jener sich eben genau so viele schwarze Subpixel befinden, wie bei einer OR-Verkettung von den n Folien in S. Das Modell für das k aus n - Schema besteht immer aus zwei Mengen von boolschen n x m - Matrizen, die wir C_0 und C_1 nennen. C_0 bezeichnet dabei die Menge der Matrizen, mit denen man ein im Originalbild weißen Pixel verschlüsseln kann und genauso benennt C_1 die Menge an Matrizen, mit denen man schwarze Pixel verschlüsselt. Die Benutzung mehrerer verschiedener Matrizen für die Verschlüsselung der Originalpixel ist wichtig, damit die Folien, auf denen das Originalbild verteilt verschlüsselt wurde, wie zufällige schwarz-weiß Muster wirken.

Wenn nun im Originalbild ein schwarzer Pixel ist, wählt man zu dessen Verschlüsselung zufällig eine Matrix aus der Menge C_1 beziehungsweise falls es sich um einen weißen Pixel handelt wählt man eine Matrix aus der Menge C_0. Damit die Entschlüsselung auch wieder gelingt, muss nun natürlich die Übereinanderlagerung der Folien in C_0 einen helleren Grauwert ergeben, der sich vom Grauwert von C_1 sichtbar unterscheidet. Und zudem darf, bei einer Übereinanderlagerung

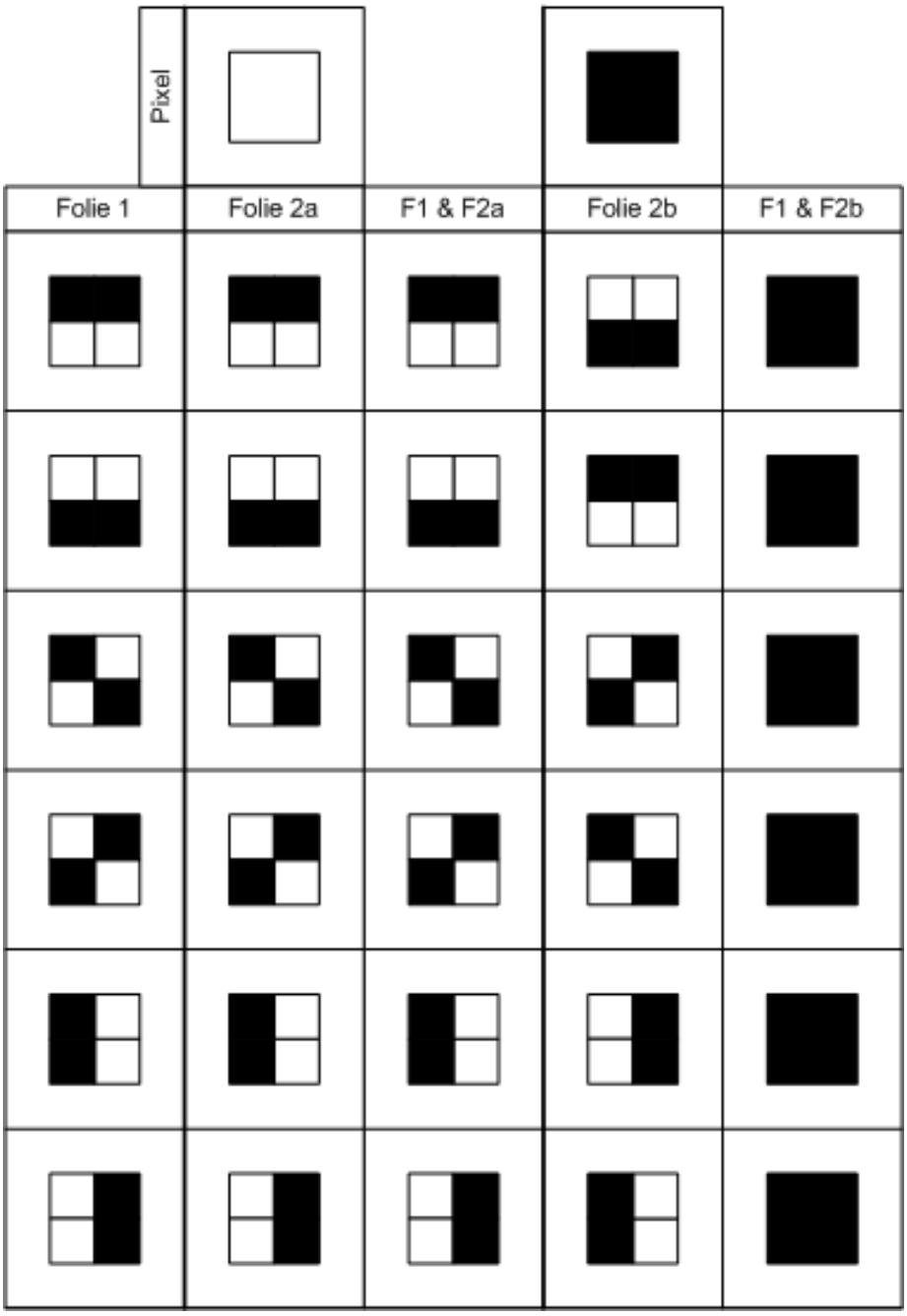

Abbildung 2: 2 aus 2 mit 4 SP

von k-1 oder weniger Folien, in keiner Weise ermittelbar sein, ob es sich dabei um eine Matrix aus C_0 oder C_1 handelt, also ob der Pixel im Originalbild weiß oder schwarz war. Ebenso sollte die Anzahl der Subpixel m so gering wie nur möglich sein, da sich der Kontrast mit der Anzahl der Subpixel verschlechtert.

Mengen C_0 und C_1 für den 2 aus 2 Fall:

$$C_0 = \{\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix},$$

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}\}$$

$$C_1 = \{\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix},$$

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}\}$$

2.4 Basismatrizen

Um effizienter mit den Matrizen in den Mengen C_0 und C_1 arbeiten zu können, verwenden wir Basismatrizen. Für den Fall 2 aus 2: Anstatt dass wir alle zwölf Möglichkeiten, wie ein schwarzer beziehungsweise weißer Pixel im Originalbild verschlüsselt werden kann, festhalten müssen, können wir auch alleine von zwei Basismatrizen S_0 und S_1 durch Permutation der Spalten auf insgesamt alle zwölf Möglichkeiten kommen. Dabei bezeichnet man mit der Basismatrix S_0 die Matrix, mit der beziehungsweise mit den Matrizen, die sich durch Permutation der Spalten der Basismatrix S_0 ergeben, die im Originalbild die weißen Pixel verschlüsseln. Gleichermaßen stellt S_1 durch Permutation der Spalten genau die Matrizen der Menge C_1 zur Verfügung. Demzufolge ist die Basismatrix S_0 beziehungsweise S_1 eine beliebige Matrix aus C_0 beziehungsweise C_1. Die Basismatrizen erweisen sich desto nützlicher, je mehr Subpixel man benötigt. Anstatt dass wir alle n x m - Matrizen, die sich in C_0 und C_1 befinden, festhalten müssen, reicht es nun aus nur jeweils eine Basismatrix S_0 für die Verschlüsselung weißer Pixel beziehungsweise S1 für schwarze Pixel festzuhalten. Durch die Permutation der Spalten, also der Anordnung der Subpixel in den verschiedenen Folien, erhält man sodann die anderen Matrizen aus den Mengen C_0 und C_1.

Die Basismatrizen S_0 und S_1 für den 2 aus 2 Fall:

$$S_0 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}, \; S_1 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

2.5 Schwellwertschema:

Während man bei dem 2 aus 2 - Verfahren noch intuitiv auf die Matrizen aus den Mengen C_0 und C_1 stoßen kann, verhält es sich schon schwieriger, wenn man ein Bild auf mehr als zwei Folien verschlüsseln will.

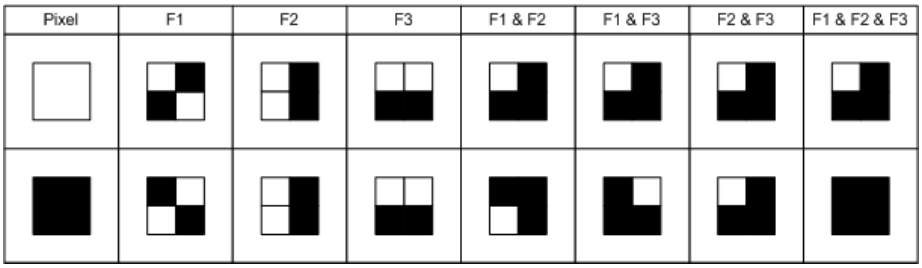

Abbildung 3: 3 aus 3 mit 4 SP

2.6 n aus n - Schema:

Bei dem n aus n - Schema darf die Originalnachricht nur dann sichtbar sein, wenn alle n Parteien ihre Folien übereinander legen und zudem darf das Originalbild, wenn weniger als n Folien übereinander gelegt werden, nicht sichtbar sein und nicht die geringste Information über das Originalbild verraten.

Konstruktion:

Die Spalten der Basismatrix S_0 sind alle boolschen n-Vektoren mit einer geraden Anzahl von Einsern und ebenso ist die Basismatrix S_1 die Matrix, deren Spalten alle boolschen n-Vektoren mit einer ungeraden Anzahl an Einsern sind. Demzufolge ist die Anzahl der Subpixel $m = 2^{(n-1)}$. Bei dem 3 aus 3 - Schema kommt man damit noch auf vier Subpixel, bei dem 5 aus 5 - Schema jedoch bereits auf 16 Subpixel. Bei einer Anzahl von 16 Subpixel ist das Originalbild bei einer Übereinanderlegung der Folien bereits sehr schlecht erkennbar. Dieses Verfahren ist dennoch optimal bezüglich des Kontrasts, d.h. es ist kein besseres Verfahren bekannt, welches mit weniger Subpixel auskommt.

Basismatrizen für 3 aus 3 Verfahren:

$$S_0 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \; S_1 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Basismatrizen für 4 aus 4 Verfahren:

$$S_0 = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}, \; S_1 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Wie man in Abbildung 3 sehen kann, wird bei Übereinanderlagerung von weniger als drei Folien keine Information preisgegeben, um feststellen zu können, ob der Pixel im Originalbild nun weiß oder schwarz ist. Das Hamming-Gewicht von zwei verschiedenen übereinander gelegten Folien ist, sowohl wenn es sich um einen weißen Pixel handelt, wie auch um einen schwarzen Pixel, immer drei. Erst bei Übereinanderlagerung aller dreier Folien, unterscheiden sich die Hamming-Gewichte und man kann feststellen, ob der Pixel im Original schwarz oder weiß ist. Dadurch ist die Sicherheit gegeben, dass weniger als n Parteien keine Information erhalten können.

2.7 Allgemeine k aus n - Schema:

Bei dem k aus n - Schema darf das originale Bild nur dann erkennbar sein, wenn mindestens k Parteien ihre Folien übereinander legen und außerdem darf das Originalbild, bei Übereinaderlegung von k-1 Folien oder weniger, nicht sichtbar sein und auch keinerlei Information über das geheime Bild preisgeben.

Konstruktion:

Existiert eine so genannte Startmatrix SM(n, l, k), so existieren auch die zwei Basismatrizen S_0 und S_1 mit $m = l \cdot 2^{(k-1)}$. Wobei die Startmatrix SM(n, l, k) eine n x l - Matrix ist, bei welcher die Einträge aus der Menge $\{a_1, \ldots, a_k\}$ sind, unter der Bedingung, dass für jeden Subset von k Zeilen mindestens eine Spalte existiert, sodass die Einträge in den k Zeilen von dieser Spalte alle verschieden sind.

Wenn es nun eine solche Matrix SM gibt, ist es leicht möglich die Basismatrix S_0 beziehungsweise S_1 zu konstruieren, indem man die Einträge $a_1, \ldots, a_k$ durch die $1, \ldots, k$-ste Zeile der Basismatrizen S_0 und S_1 vom (k, k)-Schema ersetzt.

2 aus 3 - Schema:

Wir wollen nun die Basismatrizen für den Fall 2 aus 3 finden. Dafür benötigen wir nun eine Startmatrix SM(3, l, 2), wobei $l \leq 2$ und l so klein wie möglich sein sollte, damit die Anzahl der Subpixel der Basismatrizen möglichst klein gehalten wird und ein besserer Kontrast erzielt werden kann. Wie viele Zeilen gibt es, die die obige Bedingung erfüllen mit l=2?

a_1 a_2
a_2 a_1
a_1 a_1
a_2 a_2

Da wir vier Zeilen gefunden haben, die die Bedingung erfüllen, haben wir eine Startmatrix SM(3, 2, 2) beziehungsweise ebenso eine Startmatrix SM(4, 2, 2) gefunden. Damit wissen wir außerdem, dass die Anzahl der Subpixel in beiden Fällen vier ist.

$$SM(3,2,2) = \begin{pmatrix} a_1 & a_2 \\ a_2 & a_1 \\ a_1 & a_1 \end{pmatrix}, \; SM(4,2,2) = \begin{pmatrix} a_1 & a_2 \\ a_2 & a_1 \\ a_1 & a_1 \\ a_2 & a_2 \end{pmatrix}$$

Als nächsten Schritt ersetzen wir die Einträge a_1 und a_2 jeweils durch die erste und zweite Zeile der Basismatrizen für den 2 aus 2 Fall. Durch die Ersetzung der Einträge durch die Zeilen der einen und dann der anderen Basismatrix erhält man die beiden Basismatrizen S_0 und S_1.

2 aus 2 Basismatrizen:

$$S_0 = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}, \; S_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

2 aus 3 Basismatrizen:

$$S_0 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \; S_1 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

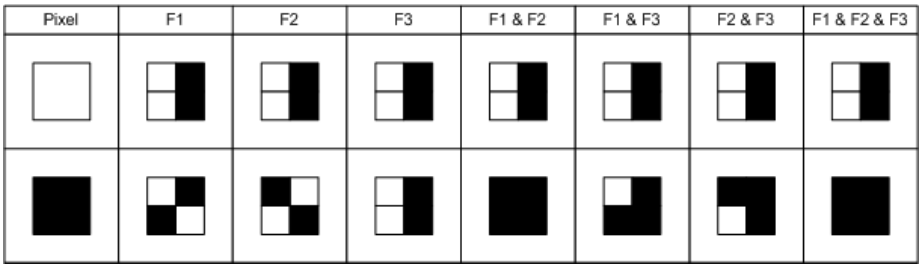

Abbildung 4: 2 aus 3 mit 4 SP

2 aus 4 Basismatrizen:

$$S_0 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \; S_1 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

Wie man in Abbildung 4 sehen kann kann man aus den einzelnen Folien noch keine Information gewinnen. Bei Übereinanderlagerung von zwei Folien ist jedoch bereits genügend Information zur Verfügung, um festzustellen, ob der Pixel im Originalbild weiß oder schwarz ist. Während das Hamming-Gewicht von zwei verschiedenen übereinander gelegten Folien bei einem weißen Pixel gleich zwei ist, ist das Hamming-Gewicht zweier Folien bei einem schwarzen Pixel drei beziehungsweise sogar einmal vier. Dadurch kann bereits durch Kombination zweier Folien das Originalbild sichtbar gemacht werden, wenn auch bei der Kombination der Folien F1 und F3 beziehungsweise der Folien F2 und F3 nicht perfektes Schwarz erreicht wird, d.h.: nicht jeder schwarze Pixel besteht nach Kombination zweier Folien aus vier schwarzen Subpixel. Dadurch ergibt sich ein Vorteil von den Parteien, die die Folien F1 und F2 besitzen, da diese zwei bei Kombination ihrer Folien perfektes Schwarz immer erreichen. Um diese Ungleichheit auszugleichen, werden in der Implementierung die Zeilen der Basismatrix zufällig vertauscht, so dass perfektes Schwarz bei jedem Pixel zwischen anderen Parteien besteht.

Ebenso kann man in Abbildung 5 die Informationssicherheit des 2 aus 4 Schema erkennen.

Da wir vorhin keine weiteren Zeilen gefunden haben, benötigen wir für die Ermittlung der beiden Basismatrizen für den Fall 2 aus 5 bereits eine Startmatrix SM(5, 3, 2).

Wie viele Zeilen können wir diesmal finden?

$a_1 \; a_1 \; a_1$
$a_2 \; a_2 \; a_2$
$a_1 \; a_1 \; a_2$
$a_1 \; a_2 \; a_1$
$a_2 \; a_1 \; a_1$
$a_1 \; a_2 \; a_2$
$a_2 \; a_1 \; a_2$
$a_2 \; a_2 \; a_1$

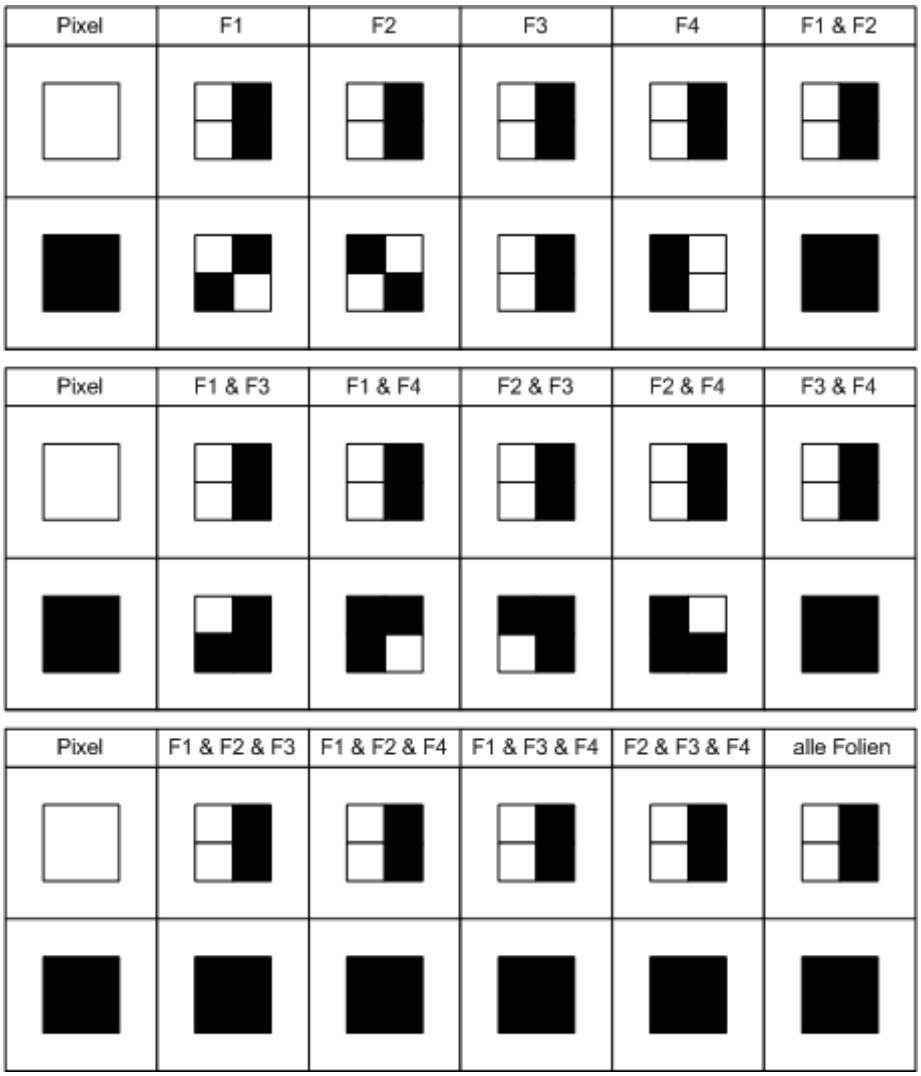

Abbildung 5: 2 aus 4 mit 4 SP

Da wir acht Zeilen finden konnten, können wir sagen, dass von (2,5) bis (2,8) man nur 6 Subpixel benötigt.

Vier Zeilen, die die obige Bedingung erfüllen:

$a_1\ a_2\ a_3$
$a_2\ a_3\ a_1$
$a_3\ a_1\ a_2$
$a_2\ a_1\ a_3$

3 aus 4 - Basismatrizen:

$$S_0 = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix},\ S_1 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Wir wollen nun die Basismatrizen für den Fall 3 aus 4 finden. Dafür benötigen wir nun eine Startmatrix SM(4, l, 3). Wie viele Zeilen gibt es, die die obige Bedingung erfüllen mit l=3?

Für den Fall 3 aus 4 benötigen wir also nach diesem Verfahren bereits zwölf Subpixel.

2.8 3 aus n - Schema

Für den Fall, dass k = 3, gibt es ein Verfahren, womit man zu einem besseren Ergebnis kommt, als mit dem oben erwähnten k aus n - Verfahren. Mit diesem Verfahren ist die Anzahl der Subpixel $m = 2 * n - 2$. Man benötigt die Matrix B, welche eine n x (n-2) - Matrix ist, die aus lauter Einsern besteht. Außerdem noch eine n x n - Einheitsmatrix. Durch Konkatenation dieser beiden Matrizen erhalten wir die Basismatrix S_1, deren Komplement die Basismatrix S_0 entspricht.

3 aus 3 - Schema:

$$B = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},\ I_{3x3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$S_0 = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix},\ S_1 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

3 aus 4 - Schema:

$$B = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix},\ I_{4x4} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$S_0 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix},\ S_1 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Damit benötigen wir für den Fall 3 aus 4 anstatt zwölf Subpixel nur sechs Subpixel und haben damit einen viel besseren Kontrast.

Bei der Übereinanderlagerung von zwei Folien von zwei Parteien, ist noch nicht sichtbar, ob der Pixel im Originalbild nun weiß oder schwarz ist, weil das Hamming-Gewicht von zwei übereinander gelegten Folien, sowohl wenn es sich im Originalbild um einen weißen Pixel als auch wenn es sich um einen schwarzen Pixel handelt gleich vier. Erst bei Übereinanderlagerung dreier Folien, unterscheiden sich die Hamming-Gewichte (falls es sich im Originalbild um einen schwarzen Pixel handelt, ist das Hamming-Gewicht fünf, bei einem weißen Pixel vier.) und man kann feststellen, ob der Pixel im Original schwarz oder weiß ist. Durch Kombination dreier Folien ist somit das Originalbild bereits sichtbar, wenn auch kein perfektes Schwarz erreicht wird. Dadurch ist die Sicherheit gegeben, dass weniger als drei Parteien keine Informationen erhalten können.

2.9 Erweiterte Visuelle Kryptographie:

Unter der erweiterten visuellen Kryptographie versteht man, dass die n Folien bereits unverdächtige Bilder enthalten, anstatt dass sie schwarz-weiß Muster sind. Durch die Kombination der Bilder ergibt sich dann das geheime Bild. Während beim normalen Verfahren nur auf die Farbe des Pixels in dem Originalbild geachtet werden muss, müssen beim erweiterten Verfahren ebenso die Farben der Pixel in den n Bildern berücksichtigt werden. Dadurch benötigt man auch anstatt zweier Basismatrizen S_0 und S_1 nun $2^{(n+1)}$ Basisindizes $S_{p_1 p_2 p_3}$ mit $p_1, p_2, p_3 \in \{s, w\}$.

3 Projektablauf

3.1 Projektauftrag

Projektname VisoCrypt - Implementierung eines Programms, mit dem die Techniken der Visuellen Kryptographie (und Steganographie) einem breiten Publikum näher gebracht werden können.

Projektauftraggeber/-leitung Ass.Prof. Dipl.-Ing. Dr. Schartner Peter

Projektmitglieder Fleiss Philipp, Pivk Mario, Winkler Nina

Startereignis Kick-Off-Meeting

Starttermin 20. Oktober 2005

Endereignis Präsentation von VisoCrypt

Präsentationstermin 4. April 2006

Ziele Folgende Ziele wurden anfangs gesetzt:

- Integration von Webcam und Drucker
- Verschlüsselung von Fotos mittels visueller Kryptographie s/w
- n-aus-k Folien-Verschlüsselung wobei:

- Verschlüsselung eines geheimen Textes bzw. Bildes in zwei unverfänglichen Bildern
- Entsprechende Darstellung für die verschlüsselten Folien und den Entschlüsselungsschritt.

Optionale Ziele Diese Ziele wurden optional dazu genomen:

- Visuelle Kryptographie: Grauwerte und Farbbilder
- Einbinden eines Spiels
- Verschlüsselung von Fotos mittels Steganographie

Projektphase/Hauptaufgaben Phasen des Projekts VisoCrypt:

- Grobspezifikation
- Implementierung der Verschlüsselungs-Methoden
- GUI designen und erstellen
- Webcameinbindung realisieren
- Druckerausgabe realisieren
- Mögliche Erweiterungen berücksichtigen
- Erstellung einer Präsentation

Zeitaufwand ca. 120h/Person (8h x 15 Wochen)

3.2 Planung

In Tabelle 1 ist der Meilensteinplan zu sehen.

Folgend kann man in Tabelle 2 die Aufgabenverteilung sehen.

Zusätzlich beträgt der Zeitaufwand für andere Aufgaben (Spezifikation, Teambesprechung/ - diskussionen, Dokumentation, Präsentationsvorbereitung) zwischen 40-60h pro Person.

3.3 Umsetzung

Zu Projektbeginn nach Erhalt der Aufgabenstellung, war eine Grobspezifikation zu erstellen. Was wollen wir mit dem Programm erreichen? Welche Funktionalitäten soll es bieten? Nach der Definition der genauen Ziele kam es zur Aufgabenteilung, diese wurde genau abgegrenzt um Erleichterung für jedes Gruppenmitglied bei der Implementierung zu gewährleisten. Man kann das Programm gut in drei Teile zerlegen, die nur durch ein paar Schnittstellen miteinander kommunizieren. Nämlich in GUI, I/O und kryptographische Algorithmen. Ab dieser Stelle konnte jeder für sich Arbeiten, da man wusste wo die Schnittstellen liegen und sich auf Konventionen einigte, der Rest lag bei jedem selbst. Erst beim Zusammenfügen der einzelnen Teile gab es rege Kommunikation zwischen den Projektmitgliedern um Testfälle für den Prototypen zu finden. Jeder sprach seine Wünsche und Anregungen in Bezug auf die Implementierung des Anderen aus, um das Programm weiter zu verbessern und zu optimieren (z.B. Designvorschläge, Skalierungen, usw.)

Meilenstein	**Basis-Plan**	**Bemerkung**	**Ist-Termin**
Projektstart	20. Oktober 2005		20.10.2005
Grobspezifikation erstellen	Anfang November		31.10.2005
Implementierung GUI	Anfang Jänner		12.01.2006
Implementierung Webcam	Anfang Jänner		12.01.2006
Implementierung Drucker	Anfang Jänner	Verzögerung wegen Bildgröße	25.02.2006
Implementierung Verschlüsselungsmethoden	Anfang Jänner		10.01.2006
Prototyp / Zwischenpräsentation	Ende Jänner / Anfang Februar		31.01.2006
Bugfixing	Mitte März		31.03.2006
Falls Projekt im Zeitplan optionale Ziele erfüllen, sonst streichen.	Anfang März	Spiel wurde gewählt	06.02.2006
Erstellung Dokumentation (Kurzes Benutzerhandbuch)	Mitte März		01.04.2006
Erstellung der Präsentation	Ende März		01.04.2006
Projektende / Endpräsentation	Ende März/ Anfang April		04.0.4.2006

Tabelle 1: Meilensteinplan unseres Projekts

Projektmitglied	**Aufgabe bei Implementierung**	**IST - Zeitaufwand (Implementierung)**
Fleiss Philipp	Implementierung der Webcam- und Druckeranbindung	120h
Pivk Mario	Implementierung der GUI	110h
Winkler Nina	Implementierung der kryptographischen Algorithmen	stunden

Tabelle 2: Aufgabenverteilung bei der Implementierung

3.3.1 GUI

Der erste Schritt beim GUI war das richtige Design zu finden mit dem alle einverstanden sind. Die beste Lösung um „visuelle Kryptographie“ einer nicht betagten Person zu präsentieren ist, unserer Meinung nach, eine Oberfläche die in drei Teile aufgeteilt ist. Das linke Drittel soll dazu dienen dem Benutzer darzustellen wie das original Bild aussieht. Im mittleren Sektor wird dann das Ergebnis der Verschlüsselung dargestellt (also nichts aussagende Folien) und im rechten Drittel wird dann der Entschlüsselungsschritt dargestellt, in dem sich die Folien aufeinander zu bewegen und bei genauer Überdeckung die Position für eine gewisse Zeit halten würden (Bild wird ersichtlich) um dann wieder von vorne zu beginnen. Es wird für alle „Tabs“ (bis auf das Spiel) die obere Leiste für Symbolbuttons reserviert z.B. Öffnen, Speichern, Verschlüsseln, Drucken, Fotografieren und Einstellungen für Parameter. Die Einfachheit der Darstellung und ein durchgehendes einheitliches Design sollte ein Hauptziel des Projektes sein. Die Benutzer werden bei falschen Eingaben mit Informationsmeldungen benachrichtigt.

3.3.2 Webcam - und Druckeranbindung

Für die Implementierung der Webcamanbindung wurde eine Recherche im Internet gemacht und man entschied sich für das Java Media Framework, um möglichst portabel zu werden. Um die Implementierung durchzuführen, wurden mehrere Tage benötigt. Wegen der Anbindung des Druckermoduls wurde ebenfalls im Internet recherchiert und diese wurde implementiert, nachdem die Webcamanbindung fertig war. Das Zusammenfügen mit der GUI klappte problemlos . Nach dem Zwischenvortrag wurde das Drucken nochmals besprochen, Verbesserungen wurden vorgenommen und fertig implementiert.

4 Dokumentation des Programms

4.1 Installation

1. Herunterladen des Java Media Framework von:

   `http://java.sun.com/products/java-media/jmf/2.1.1/download.html`

2. Öffnen von: `Start>Java Media Framework 2.1.1e>JMF Registry`

3. Lasche CaptureDevices: Button DetectCaptureDevices verwenden

4. Wenn man das JMF installiert hat und die Webcam mittels der JMF Registry erkannt wurde, kann man bei den Details den Namen und die unterstützen Auflösungen bzw Farbtiefen sehen. Sowohl den Namen der Webcam wie auch die gewünschte Auflösung und Farbtiefe kann man dann in der Datei `<KryptoToolRootDir>`/conf/webcam.conf eintragen. Dort kann man auch den Text, welcher beim Drucken unterhalb eines Bildes erscheint, verändern.

4.2 Einlesen von Bildern mit Hilfe einer Kamera

Um ein Bild mit Hilfe des Programmes in mehrere verschlüsselte Folien aufzuspalten, muss man dies natürlich zuerst einmal einlesen. Ein Bild kann entweder einfach mit dem Open- Dialog in der jeweiligen Lasche (also Split- Slide bzw Hide- Image) geöffnet werden oder eben in der ersten Lasche sozusagen live gemacht werden. Auf Abbildung 7 kann man die Webcam- Lasche sehen. In der oberen Hälfte hat man eine Toolbar mit den Funktionen zur Verfügung. In der linken Bildhälfte kann man den Live- Stream der Webcam sehen, und in der rechten Hälfte sieht man das aktuelle Bild, welches man gemacht hat. Wenn man einen Schnappschuss gemacht hat, sollte man ihn speichern oder exportieren, denn sobald man einen neuen macht, wird der alte gelöscht. Im Folgenden möchte ich die einzelnen Bedienelemente erklären. Auf Abbildung 6 kann man sehen, welche Funktionen man zum Gebrauch hat.

Abbildung 6: Toolbar in der Webcam- Lasche

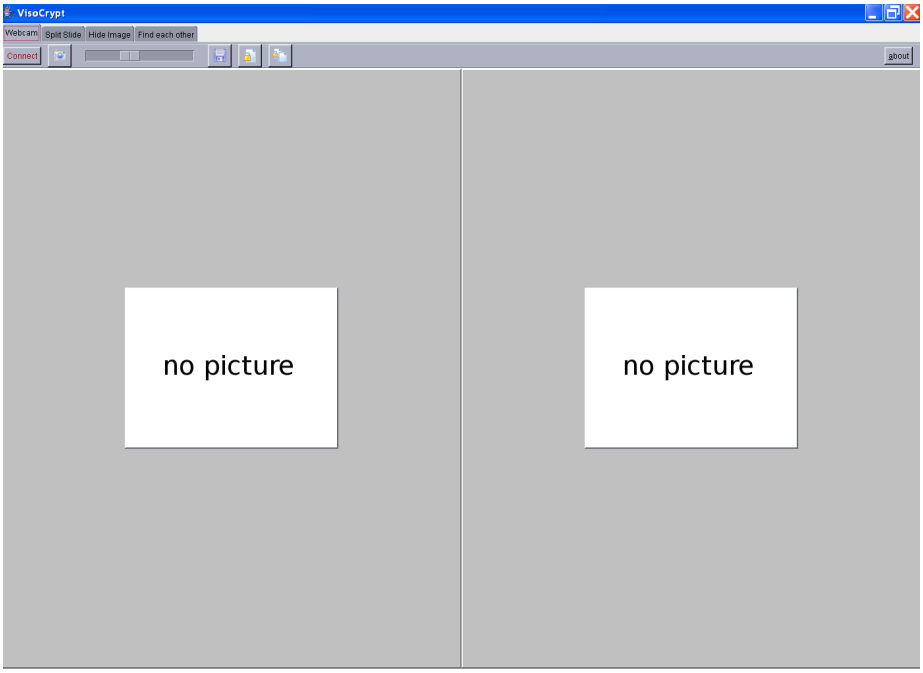

Abbildung 7: Lasche zum Einlesen von Bildern mit Hilfe einer Kamera

Connect-Button: Die vorhin erwähnten Voreinstellungen müssen erledigt sein, dann wird bei Drücken des Buttons die Verbindung zur Webcam hergestellt. Der Button wird grün und in der linken Hälfte erscheint das Live-Bild der Webcam. Bei Auftreten von Fehlern wird eine Meldung mit dem jeweiligen Problem ausgegeben.

Foto-Button: Ist nur funktionstüchtig, wenn eine Verbindung hergestellt wurde (Connect grün). Wenn das der Fall ist, wird bei Drücken des Buttons rechts eine Momentaufnahme abgebildet.

Helligkeit-Slider: Durch Verschieben des Sliders kann die Helligkeit im aufgenommenen Bild angepasst werden.

Speicher-Button: Speicherdialog öffnet sich, und das Bild wird dann im PNG- Format abgespeichert.

Export-to-Split-Slide-Button: Das Bild wir in die Kartei SS exportiert

Export-to-Hide-Image-Button: Das Bild wir in die Kartei HI exportiert

4.3 Aufspalten eines Bildes in Mehrere

Die erste Möglichkeit, welche das Programm anbietet, ist das Aufteilen eines Schwarzweiß- Bildes in eine beliebige Anzahl von Folien, welche zwischen 2 und 4 liegen kann. Diese Folien können, nachdem sie ausgedruckt wurden, übereinander gelegt werden, und man kann das ursprüngliche Bild wieder erkennen. Man kann auch einen Wert k angeben, welcher die Anzahl der Folien angibt, welche benötigt werden, um das ursprüngliche Bild zu erhalten. Wenn man beispielsweise n auf 3 einstellt und für k 2 wählt, bekommt man 3 Folien. Wenn man zwei beliebige davon übereinander legt, kann man schon das ursprüngliche Bild erkennen. Standardmäßig braucht man alle Folien zum Entschlüsseln. Auf Abbildung 8 kann man die Lasche zum Aufspalten auf mehrere Folien sehen.

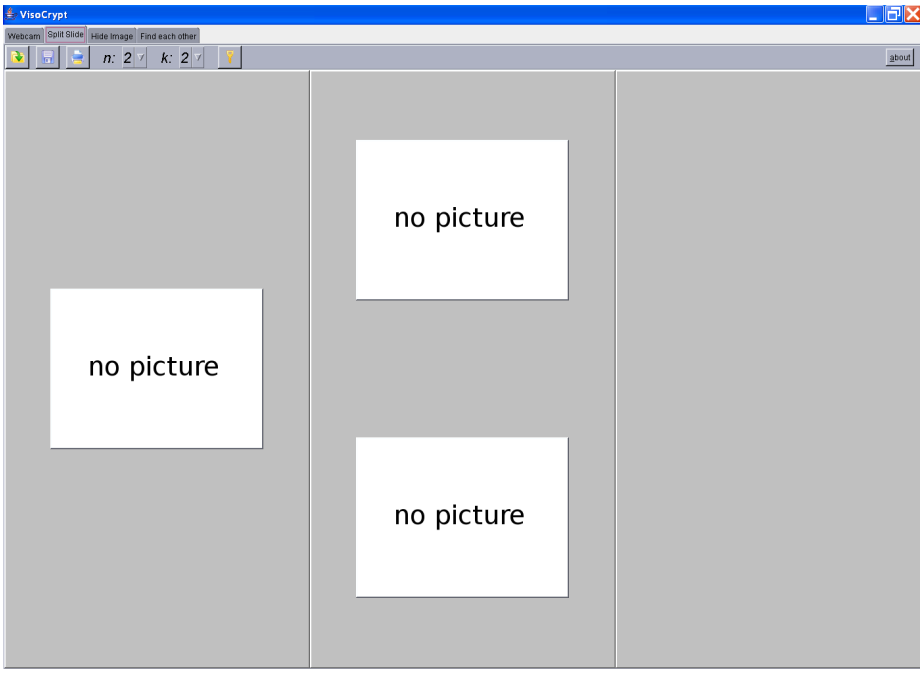

Abbildung 8: Lasche zum Aufspalten eines Bildes auf mehrere Folien

Split Slide- und Hide Image-Kartei haben die gleiche Aufteilung, welche in drei Teile geteilt ist. Im linken Drittel wird/werden die zu verschlüsselnden Bild(er) angezeigt (*Anzeige*), im mittleren Drittel werden die verschlüsselten Folien angezeigt (*Ausgabe*) und im rechten Drittel wird die Entschlüsselung visualisiert (*Präsentation*).

In der oberen Hälfte des Bildschirms ist eine Toolbar zu sehen, welche Zugriff auf die verschiedenen Funktionen der Split- Image Lasche bietet. In Abbildung 9 sehen Sie diese Toolbar.

Abbildung 9: Toolbar Split-Slide

Öffnen-Button: Öffnendialog erscheint, das geöffnete Bild erscheint in der Anzeige

Speichern-Button: Speicherdialog öffnet sich. Bilder aus der Ausgabe werden in der Form <name>-<Nr>.png abgespeichert, wobei Name der gewählte Name ist und Nr von 1 bis n geht.

Drucken-Button: Druckdialog öffnet sich. Bilder aus der Ausgabe werden ausgedruckt. Format: zwei Folien auf einer Seite (es werden nur Folien unterstützt, welche das Format 320x240 nicht überschreiten.

Auswahl von n: Anzahl der Folien die aus der Verschlüsselung entstehen können.

Auswahl von k: Anzahl der Folien die das Geheimnis auflösen können.

Verschlüsselungs-Button: Nach Wahl eines Bildes und Drücken des Buttons wird das Bild mit den angegeben Parametern n und k verschlüsselt und in der Ausgabe angezeigt, weiters wird in der Präsentation, das Entschlüsseln vorgeführt.

4.4 Verstecken eines Bildes oder Textes

Eine weitere nützliche Funktion, welche das Programm bietet, ist das Verstecken eines Bildes oder Textes in mehreren harmlosen Bildern. In der vorliegenden Version der Software kann man innerhalb von 2 Bildern ein geheimes Bild verstecken und erhält 2 Folien. Auf diesen kann man die ursprünglichen Bilder sehen, jedoch nur schwarzweiß. Auf Abbildung 10 ist die Lasche zu sehen.

Auch in dieser Lasche gibt es Buttons, deren Funktionsweise ich wiederrum in Abbildung 11 darlegen möchte.

Öffnen-Button: Öffnendialog erscheint, nach Wahl eines Bildes muss noch die Verwendung des Bildes gewählt werden (Möglichkeiten: Bild 1 bzw. 2 oder Geheimnis). Das geöffnete Bild erscheint in der Anzeige.

Speichern-Button: Speicherdialog öffnet sich. Bilder aus der Ausgabe werden in der Form <name>-<Nr>.png abgespeichert, wobei Name der gewählte Name ist und Nr von 1 bis n läuft.

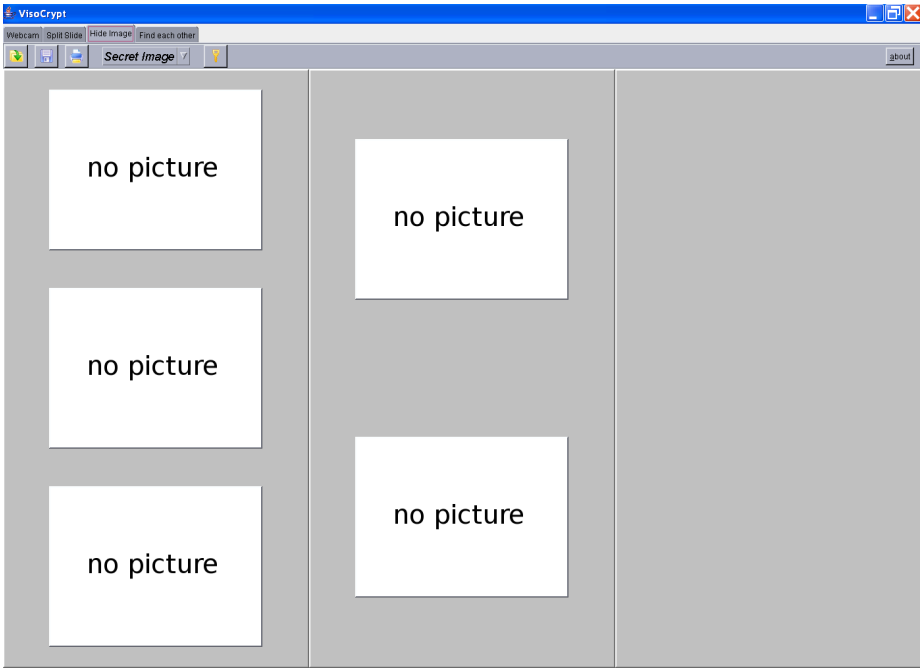

Abbildung 10: Lasche für das Verstecken von einem Bild in zwei Photos

Abbildung 11: Lasche zum Einlesen von Bildern mit Hilfe einer Kamera

Drucken-Button: Druckdialog öffnet sich. Bilder aus der Ausgabe werden ausgedruckt. Format: zwei Folien auf einer Seite (es werden nur Folien unterstützt welche das Format 320x240 nicht überschreiten.

Auswahl von Text oder Bild: Es kann gewählt werden ob ein Text oder ein Bild das Geheimnis sein sollen, bei dem Text erscheint ein Eingabefeld.

Verschlüsselungs-Button: Nach Wahl eines Bildes und Drücken des Buttons wird das Geheimnis (Bild oder Text) in die Bildern 1 und verschlüsselt und in der Ausgabe angezeigt, weiters wird in der Präsentation, das Entschlüsseln vorgeführt.

4.5 Für Gruppen: Spiel zum Suchen

Soll ein Gruppenspiel realisieren. Zuerst müssen alle teilnehmenden Spieler ein Foto machen, welches in einem Ordner gespeichert wird (in dem Ordner befinden sich nur die Fotos der Spieler). Nun wird mit dem *Choose Dir*- Button dieser Ordner ausgewählt. Nach Eingabe des Geheimtextes wird mit dem Encrypt-Button bestätigt. Nun werden aus den Bilden des Ordners jeweils zufällig zwei ausgewählt und der Geheimtext in den beiden versteckt. Bei ungerader Anzahl an Spielern wird ein Bild zweimal gewählt und als Joker bezeichnet. Mit dem Save-Button werden die verschlüsselten Bilder in den gleichen Ordner mit

dem gleichen Namen der originalen Bilder + Prefix enc_ gespeichert (der Joker wird als enc_Joker gespeichert). Der Print-Button öffnet den Druckdialog und druckt die Folien (zwei auf eine Seite) und druckt immer einen InfoText <Orndnername>-<Bildname> unter der verschlüsselten Folie mit aus.

4.6 Symbolerklärung

-  Export to Split Slide
-  Öffnen (Alt + O)
-  Verschlüsseln (Alt + E)
-  Fotographieren
-  Export to Hide Image
-  Speichern (Alt + S)
-  Drucken (Alt + P)

5 Ausblick

Nun ist es noch angebracht, Ideen für ein paar Verbesserungen zu geben. Es könnten einige Erweiterungen bei den Algorithmen vorgenommen werden. Dazu gehört das Ausweiten auf Graufstufen und / oder Farben. Auch könnte das Verstecken von einem Bild in zwei anderen Bildern so ausgeweitet werden, dass man bei Übereinanderlegung von unterschiedlichen Folien andere versteckte Bilder bekommt. Ich finde es auch wichtig, hier noch kurz anzumerken, dass das Programm einfach auf z. B.: Linux portiert werden könnte, wenn man geringe Änderungen am Quellcode machen würde. Am Schluß möchte ich noch sagen, dass das Projekt uns Spaß gemacht hat und wir den Benutzern viel Freude wünschnen .