DOI: 10.21917/ijsc.2017.0207

GENERATION OF TRANSLUCENT LANGUAGE BY SUPERIMPOSITION OPERATION

C. Annal Deva Priya Darshini[1] and V. Rajkumar Dare[2]
Department of Mathematics, Madras Christian College, India

Abstract
We have introduced a new operation called the superimposition operation. The translucent language generated by a given superimposition operation and a language L is the set of words generated by the superimposition of any two words in L. In this paper we study the properties of translucent languages. We also introduce a variant of the operation called Superimposition under control. We examine the properties of languages under this operation.

Keywords:
Translucent Words, Translucent Languages, Superimposition Under Control

1. INTRODUCTION

Research on Operation on words is a vast and independent area of study under combinatorics on words. The catenation operation is studied widely in Formal Language Theory in connection with composition of languages and their description through grammars. Several generalizations of the catenation operation like shuffle operation, insertion and deletion operations, shuffle on trajectories were introduced and studied by many authors in [6, 8]. The shuffle operation is useful in modelling parallel composition of words and languages. Theoretical generalization to the case of infinite words was studied in detail by Kadrie et al. in [1]. Another notable operation is the Collage operation on words [4]. Motivated by different real life examples, we introduced a new operation called the superimposition operation in [2]. In this paper we study the properties of languages in relation to the superimposition operation. We take our inspiration from three different real life scenarios to introduce this operation. The scope of this study is to model picture composition in Astronomy and explore the connections between Formal Languages and Astronomy.

In Astronomy, information from celestial objects is collected through special colour filters on telescopes. A colour picture is composed by overlapping multiple layers of colours using advanced image processing software. The output or final colour that we perceive is the result of the superimposition of many layers.

We model this phenomenon as an operation on words and languages. The proposed model considers each letter of the alphabet as a translucent unit like a filter so that when one is placed on top of the other, we would be able to see through. The result is a different colour or word. The rules of superimposition specify what the resulting word would be. We can define an entire class of superimposition operations on a given alphabet.

This paper is organized as follows: Section 2 gives the preliminaries required for the study. In section 3 we introduce the concept of translucent language associated with any language and study the properties. Iterated superimposition of a language is obtained by iterations of the same operation. We study the properties of iterated superimpositions also in this section. In section 4 we introduce a variation in the operation by imposing a control. We study control languages and superimposition under control in this section. The conclusion gives the relevance and future work.

2. BASIC DEFINITIONS

Let Σ be a finite alphabet and Σ^* be the collection of all words over Σ including the empty word λ . Let $\Sigma^+ = \Sigma^*-\{\lambda\}$. For $w\in\Sigma^+$, alph(w) is the elements of Σ in w. The length of a word w is $|w|$. A word is denoted by $w = w_1w_2 \ldots w_n$ where each w_i is in Σ. A word u is a factor of w if there exists x and y such that $w = xuy$. The word is a prefix or suffix of $w$if $w = uy$, $w = xu$ respectively. If $w = w_1w_2 \ldots w_n$ where each w_i is in Σ then $w^R = w_nw_{n-1} \ldots w_1$. A language L is a set of words or a subset of Σ^*. L^c denotes the complement of a language.

The shuffle operation [8], denoted by $\Diamond$, is defined recursively by,

$$(au\Diamond bv) = a(u\Diamond bv) \cup b(au\Diamond v), \text{ and } (u\Diamond\lambda) = (\lambda\Diamond u) = \{u\}$$

where, $u,v \in \Sigma^*$ and $a,b \in \Sigma$.

The shuffle of two languages L_1 and L_2 is $L_1\Diamond L_2 = \bigcup_{u\in L_1, v\in L_2} u\Diamond v$.

Example: $(ab\Diamond bc) = \{abbc, abcb, babc, bacb, bcab\}$.

The Collage operation on words is defined in [4] as: Given a subset $W \subseteq \Sigma^*$ of patches, the operation of Collage consists of producing words in $(\Sigma\cup\{\Delta\})^*$, (Δ is a new symbol not in Σ), by starting with a word of the form Δ^m and then repeatedly replacing random factors of the current word with elements of W. A word thus obtained is called a Collage of W. Define $C^0(W) = \Delta^*$ and for all $k \geq 0, C^{k+1}(W) = \{w[i \leftarrow z] \mid w \in C^k(W), z \in W, 1 \leq i \leq |w| - |z| + 1$

Example: Consider $n = 11$ and assume the words *aba, bbbbc, ca, abaabcab* belong to the subset W and are placed respectively at the positions 2, 4, 10 and 1 in that order. The resulting word is at the top of the following Table.1.

Table.1. Collage of W

a	*b*	*a*	*a*	*b*	*c*	*a*	*b*	-	*c*	*a*
a	*b*	*a*	*a*	*b*	*c*	*a*	*b*			
									c	*a*
			b	*b*	*b*	*b*	*c*			
	a	*b*	*a*							
1	2	3	4	5	6	7	8	9	10	11

Position 4 is covered by the occurrences *aba, bbbbc* and *abaabcab.* Position 9 is covered by no occurrence and position 2 is covered by the occurrences of *aba* and *abaabcab.*

3. TRANSLUCENT LANGUAGES

We introduce the concept of superimposition operation and translucent languages generated by a language. This language has some interesting properties.

We recall the definition of superimposition operation introduced in [2].

For $a,b \in \Sigma$, $a \otimes b = c$ where c is also in Σ is called the superimposition operation. This operation can be extended to Σ^*.

Definition 3.1

Let $u, v \in \Sigma^+$ and $u = u_1u_2...u_n$, $v = v_1v_2...v_m$ where $u_i, v_j \in \Sigma$, for $i = 1,2,...,n$, $j = 1, 2,...,m$.

a) If $|u| = |v|$, then $u\otimes v = (u_1\otimes v_1)(u_2\otimes v_2)...(u_n\otimes v_n)$

b) If $|u| > |v|$, and if $u = u'u''$ where $|u'| = |v| = m$,

then $u\otimes v = (u_1\otimes v_1)(u_2\otimes v_2)\ldots(u_m\otimes v_m).\ u''$.

c) If $|u| < |v|$, and if $v = v'v''$ where $|v'| = |u| = n$,

then $u\otimes v = (u_1\otimes v_1)(u_2\otimes v_2)...(u_n\otimes v_n).v''$

d) For $u \in \Sigma^*$, $u\otimes\lambda = \lambda\otimes u = u$.

Given an alphabet, it is possible to define a class of superimposition operations on the alphabet. If $X = \{a,b\}$, it is possible to specify 16 different rules. The word z generated from two words u and v is unique and is called a translucent word obtained from u and v. We use a representation of the operation in the form of a table.

Example 3.1

Let $\Sigma = \{b,w,g\}$ and the operation $\otimes$ given by the following Table.2.

Table.2. Superimposition rule

$\otimes$	b	w	g
b	b	b	b
w	b	w	g
g	b	g	g

Let $x = (bw)^5$, $y = (wb)^5$ be words in Σ^* then $x\otimes y = b^{10}$.

Definition 3.2

Let $a,b,c \in\Sigma$. The operation $\otimes$ is

a) associative if $a\otimes(b\otimes c)=(a\otimes b)\otimes c$,

b) commutative if $b\otimes a = a\otimes b$,

c) Idempotent if $a\otimes a = a$.

Some simple results of the operation are immediate.

Let x, y, z be words in Σ^+. Then

a) $|x\otimes y| = \max\{|x|,|y|\}$,

b) $x\otimes y = y\otimes x$ if and only if $\otimes$ is commutative,

c) $x\otimes(y\otimes z)=(x\otimes y)\otimes z$ if and only if $\otimes$ is associative

The operation can be extended to languages.

Definition 3.3

Let $L_1, L_2 \subseteq \Sigma^+$ then the superimposition of L_1 and L_2 is $L_1\otimes L_2 = \bigcup_{u\in L_1, v\in L_2} u\otimes v$. For a given language L,

$$L^0_\otimes = \lambda, L^1_\otimes = L, L^{i+1}_\otimes = L^i_\otimes \otimes L.$$

Then the star closure of a language is $L^*_\otimes = L^0_\otimes \cup L^1_\otimes \cup L^2_\otimes.....$ and the positive closure is $L^+_\otimes = L^1_\otimes \cup L^2_\otimes.....$ for a given superimposition operation. The superimposition of two languages is always a non-empty language.

Definition 3.4

For a given language L, the translucent language generated by L under the superimposition operation $\otimes$ is given by $T(L,\otimes) = \{w \in \Sigma^+ / w = u\otimes v, u,v\in L\}$.

Example 3.2

Let $\Sigma = \{a,b\}$ and let the operation $\otimes$ be specified by Table.3.

Table.3. Superimposition rule

$\otimes$	a	b
a	a	b
b	b	b

Let L be the set of words having exactly one a. The translucent language generated is $T(L,\otimes) = L\cup\{b^n / n>0\}$.

The translucent language generated by a language is either itself or different from the language. However, there are some special cases where the translucent language generated is identical to the language from which it is generated. We give an example where the translucent language is the same as the language from which it is generated.

Example 3.3

For the unary alphabet $\Sigma = \{a\}$, if $L = \{a^n / n > 0\}$ then $T(L,\otimes) = L$

We give an illustration of the operation as a simulation on how a colour image is produced.

Generation of a colour picture involves the use of two main colour spaces – RGB and CMYK. The RGB colour model related to the way we perceive colour with the r, g, b receptors in our retina. RGB uses additive colour mixing and is the basic colour model used in television or any other medium that projects colour with light. The secondary colours of RGB – cyan, magenta and yellow – are formed by mixing two of the primary colours and excluding the third colour. The four colour CMYK model used in printing lays down overlapping layers of transparent inks to form a colour picture. In the following example we generate a transcluscent language by the superimposition of colours.

Example 3.4

Let $\Sigma = \{r, g, b, c, y, m, k\}$ denote the colours

r – red

g – green

b – blue

c – cyan

y – yellow

m – magenta

k – black

and the superimposition operation be given by Table.4.

Let $L = \{(ymy)^p c^{2q}, p,q > 0\} \cup \{(mym)^p (y^q m^q), p,q > 0\}$

then $T(L,\otimes) = L \cup \{r^p (gb)^q), p,q > 0\}$.

We note that the superimposition operation on the colours is a commutative and associative operation.

Table.4. Superimposition rule on colours

$\otimes$	*r*	*g*	*b*	*c*	*y*	*m*	*k*
r	*r*	*k*	*k*	*k*	*r*	*r*	*k*
g	*k*	*b*	*k*	*g*	*g*	*k*	*k*
b	*k*	*k*	*b*	*b*	*k*	*b*	*k*
c	*k*	*g*	*b*	*c*	*g*	*b*	*k*
y	*r*	*g*	*k*	*g*	*y*	*r*	*k*
m	*r*	*k*	*b*	*b*	*r*	*m*	*k*
k	*k*	*k*	*k*	*k*	*k*	*k*	*k*

A black and white picture would look something like this.

Example 3.5

Let, Σ be the alphabet {■ □ ▨}

where, ■ □ ▨ denote the colours black, white and gray respectively and the superimposition operation be given by Table.5.

Let *L* be the language, {(■□■□■□■□)^n, $n > 1$}

then $T(L,\otimes) = L$.

Table.5. Superimposition rule on black and white colours

$\otimes$	■	□	▨
■	■	■	■
□	□	■	□	▨
▨	▨	■	▨	▨

Proposition 3.1

If $L \subseteq \Sigma^n$ is closed with respect to a superimposition ⊗ then $T(L,\otimes)$ is also closed with respect to the superimposition operation.

Proof: Let $w_1, w_2 \in T(L,\otimes)$. Since $w_1 \in T(L,\otimes)$, we have $w_1 = x_1 \otimes y_1$ for some $x_1, y_1 \in L$. Similarly, $w_2 = x_2 \otimes y_2$ for some $x_2, y_2 \in$ L. Since *L* is closed, $x_1 \otimes y_1 = x \in L$ and $x_2 \otimes y_2 = y \in L$. Now, $x \otimes y = z \in T(L, \otimes)$ by the definition of translucent languages. Hence $T(L,\otimes)$ is closed.

A language *L* is commutative [6] if for any $w \in L$ contains all the words obtained from *w* by arbitrarily permuting its letters. For a word $u = a_1 a_2 \ldots a_k \in \Sigma^*$, $k \geq 0$, we define $com(u) = \{a_{s(1)} a_{s(2)} \ldots a_{s(k)} | s$ a permutation of $\{1,\ldots,k\}\}$ that is, $com(u)$ contains all the words obtained by arbitrarily permuting the letters of *u*. If $L \subseteq \Sigma^*$ then $com(L) = \bigcup_{u \in L} com(u)$. *L* is commutative if and only if $L = com(L)$.

Proposition 3.2

If $L \subseteq \Sigma^+$ is a commutative language then $T(L, \otimes)$ is also a commutative language.

Proof: Let $w \in T(L, \otimes)$. Then $w = x \otimes y$ for some $x,y \in L$. Since *L* is commutative, $com(x), com(y) \in L$. This implies $com(x) \otimes com(y) \in \mathrm{T}(L,\otimes)$ for all possible permutations of $\{1,2,\ldots,n\}$. Further, $com(x) \otimes com(y) = com(x \otimes y)$ for each permutation.

Proposition 3.3

The translucent language $T(\Sigma^n,\otimes)$ is always a commutative language.

Proof: By definition, Σ^n is commutative and $T(\Sigma^n,\otimes) = \Sigma^n$.

Note that $T(L,\otimes) = L^2_{\otimes}$.

Definition 3.5

The iterated superimposition of two languages is given by $L_1 \otimes^* L_2 = \bigcup_{n=0}^{\infty} (L_1 \otimes^n L_2)$, where

$$L_1 \otimes^0 L_2 = L_1 \text{ and } L_1 \otimes^{n+1} L_2 = (L_1 \otimes^n L_2) \otimes L_2 .$$

We say that a superimposition operation is neutral if

a) The superimposition operation is given by the following rules: $a \otimes a = a \otimes b = b \otimes a = a$, $b \otimes b = b$,

b) $a \otimes a = a$, $a \otimes b = b$, $b \otimes a = a$, $b \otimes b = b$.

The neutral superimposition leaves the word unaltered or hides the word completely. We call the superimposition operation given by rule (i) as Type I neutral operation and the rule (ii) as Type II neutral operation.

Proposition 3.4

If ⊗ is a neutral operation of Type I or Type II then $T(L, \otimes) = L$ for any $L \subseteq \Sigma^+$.

Proposition 3.5

If ⊗ is a neutral operation then we have

a) $L_1 \otimes^n L_2 = L_1$ for any $L \subseteq \Sigma^+$ if the operation is of Type I and

b) $L_1 \otimes^n L_2 = L_2$ for any $L \subseteq \Sigma^+$ if the operation is of Type II.

Proposition 3.6

If ⊗ is a neutral operation then we have

a) $T(L_1 \otimes^n L_2, \otimes) = L_1$ for any $L \subseteq \Sigma^+$ if the operation is of Type I and

b) $T(L_1 \otimes^n L_2, \otimes) = L_2$ for any $L \subseteq \Sigma^+$ if the operation is of Type II.

Proof follows from propositions 3.4 and 3.5.

Proposition 3.7

If ⊗ is idempotent then we have,

a) $L \subset T(L, \otimes)$ for any $L \subseteq \Sigma^+$,

b) $L \otimes^n L = L$ for any $L \subseteq \Sigma^+$.

Proof: If $x \in L$ then $x \otimes x = x$ since the operation ⊗ is idempotent. This implies $x \in \underline{T}(L,\otimes)$.

4. SUPERIMPOSITION UNDER CONTROL

In this section, we introduce a variant of this operation where we restrict the superimposition of two words using a control language. Here, we subdivide both the words into factors using the control language as a guide and then allow the factors to superimpose.

We introduce a new alphabet $V = \{f, s\}$ to define the operation. Let $c \in V^*$ be the control word and $C \subseteq V^*$ be a control language.

Definition 4.1

Let $x = x_1,x_2,\ldots x_m \in \Sigma^*$ and $y = y_1,y_2,\ldots y_n \in \Sigma^*$ for $x_i, y_i \in \Sigma$, $i = 1,2,\ldots m$, $j = 1,2,\ldots n$.

Let $c = z_1^{n_1} z_2^{n_2} \ldots z_k^{n_k} \in V^*, z_i \in V, n_i \in \mathrm{N}$.

If $|c|_f = |x|, |c|_s = |y|, |c| = |x| + |y|$, we define the superimposition of x and y controlled by $c \in V^*$ as

$$x \otimes_c y = \begin{cases} (x_1x_2\ldots x_{n_1}) \otimes (y_1y_2\ldots y_{n_2}) \ldots.., z_1 = f \\ (y_1y_2\ldots y_{n_1}) \otimes (x_1x_2\ldots x_{n2_1}) \ldots.., z_1 = s \\ \varphi, otherwise \end{cases}$$

Each $z_i \in V$ is repeated a certain number of times in c. The number of repetitions is counted by the indices n_1, n_2, … If z_1 is f then z_2 is s. In such a case, the first n_1 letters of the first word is superimposed on the first n_2 letters of the second word. The z_i alternate between f and s.

Hence it is enough to check the first letter of C. The superimposition is repeated until all the letters of the control word are exhausted. The word generated is unique and is of length $max(n_1, n_2) + max(n_3, n_4) + \ldots max(n_{k-1}, n_k)$.

Example 4.1

Let $x = a_1a_2\ldots a_8 \in \Sigma^*$ and $x = b_1b_2\ldots b_5 \in \Sigma^*$, $c = f^3s^2f^3sfsfs$.

Note that $|c|_f = |x|, |c|_s = |y|, |c| = |x| + |y|$. The word x is factorised as $x = u_1u_2u_3u_4$ and $y = v_1v_2v_3v_4$ using c as a control. We have,

$$u_1 = a_1a_2a_3,\ u_2 = a_4a_5a_6,\ u_3 = a_7,\ u_4 = a_8,$$
$$v_i = b_1b_2,\ v_2 = b_3,\ v_3 = b_4,\ v_4 = b_5.$$

Therefore,

$$x \otimes_c y = (a_1a_2a_3 \otimes b_1b_2)(a_4a_5a_6 \otimes b_3)(a_7 \otimes b_4)(a_8 \otimes b_5)$$

Definition 4.2

If c is a control language over V^* then $x \otimes_C y = \bigcup_{c \in C} x \otimes_c y$.

If c is the empty set then $x \otimes_c y = \varphi$.

Also if $L_1, L_2 \subseteq \Sigma^*$ then $L_1 \otimes_C L_2 = \{x \otimes_C y / x \in L_1, y \in L_2\}$.

Intuitively, a control language gives us a guide to subdivide two words in a particular way and to superimpose the subwords. The factorization of x and y into subwords using a control word gives us an alternate definition of superimposition under control.

Definition 4.3

Let us denote the run of a letter in $c \in V^*$ be the number of consecutive occurrences of the letter in c.

For example, if $c = f^3s^2$ then $run(f) = 3$ and $run(s) = 2$.

If c is a word with more than one non – consecutive occurrence of the letter f or s then we denote the runs as $run_i(f)$.

If $c = f^3s^2f^3sfsfs$ then $run_1(f) = 3$, $run_2(f) = 3$, $run_3(f) = 1$, $run_4(f) = 1$, $run_1(s) = 2$, $run_2(s) = run_3(s) = run_4(s) = 1$.

Then if $\sum_i run_i(f) = |x|$ and $\sum_j run_j(s) = |y|$ we factorise $x = u_1u_2\ldots$, $y = v_1v_2\ldots$ where $|u_i| = run_i(f)$ and $|v_j| = run_j(s)$ for $i, j = 1, 2, \ldots..$

In such a case, if $t \in V$ denotes f or s and $c = tV^*$ we have

$$x \otimes_c y = \begin{cases} (u_1 \otimes v_1)(u_2 \otimes v_2)\ldots.., t = f \\ (v_1 \otimes u_1)(v_2 \otimes u_2)\ldots.., t = s \\ \varphi, otherwise \end{cases}$$

Example 4.2

Let $\Sigma = \{a,b\}$ and $V = \{f, s\}$. Let $c \in V^{14}$ and let $x, y \in \Sigma^7$ be words on Σ. Let $x = abaabab$, $y = abbbaab$.

Then,

a) if $c = f^7s^7$ we have $x \otimes_c y = x \otimes y$

b) if $c = (fs)^7$ we have $x \otimes_c y = x \otimes y$

c) if $c = f^7$ we have $x \otimes_c y = x$

d) if $c = s^7$ we have $x \otimes_c y = y$

e) if $c = s^7f^7$ we have $x \otimes_c y = y \otimes x$.

Definition 4.4

Let C be a control set. We say that C is commutative if and only if the operation $\otimes_c$ is commutative, that is, $x \otimes_c y = y \otimes_c x$ for all $x,y \in \Sigma^*$.

Let $\aleph$ be the family of all commutative sets of control words.

Proposition 4.1

If $(C_i)_{i \in I}$ is a family of control languages such that (C_i) is a commutative control language for all $i \in I$ then their intersection $C' = \bigcap_{i \in I} C_i$ is also a commutative control language.

Proof:

Let $u,v \in \Sigma^*$ and $w \in u \otimes_{C'} v$. Then it follows that $w \in u \otimes_{C_i} v$ for all $i \in I$. But, each (C_i) is commutative and hence $w \in v \otimes_{C_i} u$ for all $i \in I$. Therefore, $w \in v \otimes_{C'} u$. Thus, we have $u \otimes_{C'} v = v \otimes_{C'} u$ which implies that $C' = \bigcap_{i \in I} C_i$ is also a commutative control language.

Definition 4.5

Let C be a control language. The commutative closure of C denoted by $\overline{C}$ is given by $\overline{C} = \bigcap_{C \subseteq C', C' \in \aleph} C'$.

Corollary: For all $C \subseteq \{f,s\}^*$, $\overline{C}$ is a commutative control language.

Remark: $\overline{C}$ is the smallest control language that contains C.

Definition 4.6

A control language C is associative if and only if $\otimes_c$ is associative. We have $x \otimes_c (y \otimes_c z) = (x \otimes_c y) \otimes_c z$ for all $x,y,z \in \Sigma^*$.

Proposition 4.2

If $\left(C_i\right)_{i\in I}$ is a family of control languages such that (C_i) is an associative control language for all $i \in I$ then their intersection $C'' = \bigcap_{i\in I} C_i$ is also an associative control language.

Proof: Let $x, y, z \in \Sigma^*$ and let $w \in x \otimes_{C''} (y \otimes_{C''} z)$. Then it follows that $w \in x \otimes_{C_i} (y \otimes_{C_i} z)$ for all $i \in I$. But, each $\left(C_i\right)$ is associative and hence $w \in (x \otimes_{C_i} y) \otimes_{C_i} z$ for all $i \in I$. Therefore, $w \in (x \otimes_{C''} y) \otimes_{C''} z$. Thus, we have $x \otimes_{C''} (y \otimes_{C''} z)$ $= (x \otimes_{C''} y) \otimes_{C''} z$ which implies that $C'' = \bigcap_{i\in I} C_i$ is also an associative control language.

Definition 4.7

Let C be a control language. The associative closure of C denoted by $\overline{\overline{C}}$ is $\overline{\overline{C}} = \bigcap_{c \subseteq C'', C' \in A} C''$ where A is the family of all associative control languages.

Proposition 4.3

The associative closure $\overline{\overline{C}}$ is also an associative control language.

Proposition 4.4

Let $L_1, L_2 \subseteq \Sigma^*$ be regular languages. Then $L_1 \otimes L_2$ is regular.

Proof: We give the automata to prove that the superimposition of two regular languages is also regular. Let A_1 and A_2 be two finite automata accepting the languages $L(A_1)$ and $L(A_2)$. Then we can find an automaton A such that $L(A) = L(A_1) \otimes L(A_2)$. Let $A_i = (Q_i, \Sigma, \delta_i, q_0^i, F)$ for i = 1, 2 be the two finite automata accepting languages $L(A_1)$ and $L(A_2)$. We construct the automaton $A = (Q, \Sigma, \delta, q_0, F)$ as $Q = Q_1 \times Q_2, F = F_1 \times F_2, q_0 = (q_0^1, q_0^2)$ and $\delta((q_i^1, q_j^2), (a, b)) = ((q_k, q_l), c)$ where, $\delta_1(q_i^1, a) = q_k$, $\delta_2(q_j^2, b) = q_l$ and $a \otimes b = c$ is the rule of superimposition specified.

The automaton A accepts an input w if and only if A_1 accepts w_1 and A_2 accepts w_2 where the transition rules are $\delta((q_i^1, q_j^2), (w_1, w_2)) = ((q_k, q_l), w), (q_k, q_l) \in F$ for $\delta_1(q_i^1, w_1) = q_k$, $q_k \in F_1$ and $\delta_2(q_j^2, w_2) = q_l, q_l \in F_2$ and $w_1 \otimes w_2 = w$ .Then $L(A) =$ $L(A_1) \otimes L(A_2)$.

Proposition 4.5

For every language, $L \subseteq \Sigma^*$, $\Sigma = \{a,b\}$ there exists a control language C such that $L = a^* \otimes_c b^*$.

Proof: Let φ be a morphism, $\varphi(a)=f$, $\varphi(b)=s$ and let $C = \varphi(L)$ then $L = a^* \otimes_c b^*$.

Theorem 4.1

Let $V = \{f, s\}$. Let $C \subseteq V^*$ be a control language, then the following are equivalent:

a) For all regular languages $L_1, L_2 \subseteq \Sigma^*$, the language $L = L_1 \otimes_c L_2$ is a regular language,

b) C is a regular language.

Proof: To prove (i) implies (ii). Assume $L_1 = f^*$ and $L_2 = s^*$ then $L_1 \otimes_c L_2 = C$. Therefore, C is a regular language.

To prove (ii) implies (i), Assume that C is a regular language. Let L_1, L_2 be two regular languages over the same alphabet Σ. Let $A_i = (Q_i, \Sigma, \delta_i, q_0^i, F)$ for i = 1, 2, be finite automata such that $L(A_i)$ = L_i for i = 1,2. Also, let $A_c = (Q_c, V, \delta_c, q^c, F_c)$ be an automaton such that $L(A_c)$ = C. Define an automaton $A = (Q, \Sigma, \delta, Q_o, F)$ such that. $L(A) = L_1 \otimes_c L_2$ The input is accepted by if and only if A_1, A_2, A_c are in accepting states. Then $L_1 \otimes_c L_2$ is a regular language. We have,

$$Q = Q_1 \otimes Q_c \otimes Q_2 \text{ and } Q_o = \{q_o^1, q_o^2, q_o^c]$$

where, $F = F_1 \otimes F_c \otimes F_2$ and $\delta(q_1^1, c, q_2^2, a, b) = (q_1, q_2, c)$, $\delta_1(q_1^1, f, a) = q_1$, $\delta_2(q_2^2, s, b) = q_2$, $a \otimes_c b = c$, $c \in V^*$.

Define the transformation σ:{*f*,*s*}→{*f*,*s*}* as σ(*f*) = *s*,σ(*s*) = *f*. Then we have the following property of superimposition under control.

Proposition 4.6

Let $L_1, L_2 \subseteq \Sigma^*$ and $C \subseteq V^*$ be any two languages and a control language. Let $x_1 \in L_1$, $x_2 \in L_2$ and $c_1, c_2 \in C$. We have $x_1 \otimes_c x_2 = x_2 \otimes_c x_1$ if and only if $c_2 = \sigma(c_1)$.

Definition 4.8

A control language C is commutative if and only if $C = \sigma(C)$.

Proposition 4.7

If C is a regular control language then it is decidable whether or not C is commutative.

Proof: If C is a regular language then $\sigma(C)$ is also regular. Hence the equality, $C = \sigma(C)$ is decidable.

5. CONCLUSION

We have introduced the translucent language associated with a language generated by the operation and studied the properties. The natural extension to arrays is an obvious direction in which the work can proceed. Expressions involving superimpositions were introduced in [2]. This study can be taken further.

REFERENCES

[1] Ahmad Kadrie, V. Rajkumar Dare, D.G. Thomas and K.G. Subramanian, "Algebraic Properties of the Shuffle Over ω-Trajectories", *Information Processing Letters*, Vol. 80, No. 3, pp. 139-144, 2001.

[2] C. Annal Deva Priya Darshini and V. Rajkumar Dare, "Superimposition Operations on Words", *Proceedings of National Conference on Mathematics and Computer Applications*, pp. 48-52, 2015.

[3] Christian Choffrut and Juhani Karhumaki, "Combinatorics of words in Handbook of Formal Languages", Vol. 1, *Handbook of Formal Languages, Springer*, pp. 329-438, 1997.

[4] Christian Choffrut and Berke Durak, "Collage of Two-Dimensional Words",*Theoretical Computer Science*, Vol. 340, No. 2, pp. 364-380, 2005.

[5] J.E. Hopcroft and J.D. Ullman, "*Introduction to Automata Theory, Languages and Computation*", Narosa Publishing House, 1979.

[6] Masami Ito, Lila Kari and Gabriel Thierrin, "Shuffle and Scattered Deletion Closure of Languages", *Theoretical Computer Science,* Vol. 245, No. 1, pp. 115-133, 2000.

[7] Jean Berstel and Dominique Perrin, "The Origins of Combinatorics of Words", *European Journal of Combinatorics,* Vol. 28, No. 3, pp. 996-1022, 2007.

[8] A. Mateescu, G. Rozenberg and A. Salomaa, "Shuffle on Trajectories: Syntactic Constraints", *Theoretical Computer Science*, Vol. 197, No. 1-2, pp. 1-56,1998.

[9] J.E. Tremblay and R. Manohar, "*Discrete Mathematical Structures with Applications to Computer Science*", Tata McGraw Hill, 1997.