Superposition of Rectangles with Visibility Requirement: A Qualitative Approach

Takako Konishi
Graduate School of Science & Technology
Kwansei Gakuin University
2-1, Gakuen, Sanda, 669-1337, JAPAN
Email: t.konishi@kwansei.ac.jp

Kazuko Takahashi
School of Science & Technology
Kwansei Gakuin University
2-1, Gakuen, Sanda, 669-1337, JAPAN
Email: ktaka@kwansei.ac.jp

***Abstract*—This paper discusses the superposition of qualitative rectangles so that some parts are visible and other parts are hidden based on the user's requirements. Qualitative rectangles are rectangles whose size and edge ratios are not fixed. We propose a symbolic representation of the target objects and discuss superposition on this representation. The operations of superposing two rectangles can be defined either by superposing some specified parts of rectangles or by embedding one rectangle into part of the other rectangle. We investigate the conditions under which such a superposition succeeds as well as the manner in which such superposition occurs. We developed an algorithm for superposing multiple qualitative rectangles and implemented it.**

Keywords-qualitative knowledge representation; superposition; rectangle packing; spatial database

I. Introduction

Personal computer users commonly open multiple windows. When many windows are opened on a narrow screen, the most newly opened window usually appears in the foreground, sometimes hiding important parts of previously opened windows. Users must frequently resize windows or move a window to the foreground to ensure that important parts are visible. Many users find this process annoying; it would be more convenient if windows were positioned automatically so that important or necessary parts remain visible under the condition that these parts are specified in advance. Moreover, considering the limited screen space, it would be more efficient if less important parts of windows are hidden.

In general, researchers have investigated the efficient placement of objects as a type of packing problem for which an optimal solution can be determined [1]. They have focused on several application areas, such as VLSI design [2] and label-placement problems [3], [4]. The objective of these studies has been to determine how multiple objects are located in a two-dimensional plane under circumstances not involving superposition, which differs from our objective of determining placement involving superposition. Therefore, we cannot directly apply the previously developed algorithms for general placement problems. To the best of the authors' knowledge, no study has been performed on the location of objects involving superposition.

In this study, we discuss rectangle placement with superposition. We treat rectangles using qualitative representation: their sizes and the ratios of their edges are unfixed. In each rectangle, the desired visible part is specified. We discuss a manner of superposing them so that all desired visible parts are in the foreground and all desired hidden parts are in the background. Figure 1 illustrates several examples. Assume that three rectangles A, B, and C are given with a requirement of visibility specified by a user. The white indicates the parts that one wants to be visible, and the black indicates the parts that one wants to be hidden. In this figure, (a), (b), and (c) are successful cases, whereas (d) is not. In (c), first reduce B's width to fit the vertically-long-size subpart of the black part of A, then C is put on the black part in the lower left part of the resultant figure. In (d), visible black parts remain after superposing A and B cannot be hidden by C in any superposition of A and B. In this paper, we show how we evaluate the success of superposition and placement in these cases.

Note that the sizes or ratios of edges can change during superposing. We take a qualitative approach. One reason for this is that it enables symbolic handling of objects. In general, spatial data can be inconveniently large to store and handle. Symbolic handling reduces this computational complexity. Another reason is that it is enough to know the relative positional relationship of objects on a two-dimensional plane and their foreground/background relationship, ignoring the exact size or position of each object. Such an idea is considered to be a type of qualitative spatial reasoning (QSR) in the field of artificial intelligence [5], [6], [7], [8].

Our goal is not to find an optimal solution to the packing problem, but to investigate methods for symbolic treatment of spatial data and to develop possible application areas of qualitative spatial reasoning. An earlier version of our work was reported in [9], [10]. The present study expanded on our previous research, and this paper provides a more detailed discussion.

In our QSR approach, target rectangles are divided into nine types depending on the specified visibility pattern. We define a unique symbolic representation for each type and investigate superposition operations using these representa-

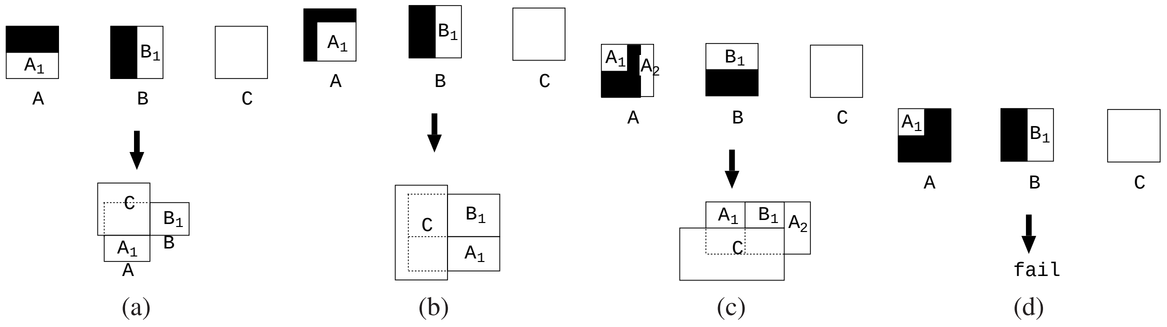

Figure 1. Examples of superposing rectangles

tions. Superposition succeeds if any rectangle is not placed on the desired visible parts of another rectangle.

We focus on two types of superposition. *puton* is an operation that corresponds to superposition of specified parts of two rectangles. *embed* is an operation that corresponds to the embedding of a whole part of one rectangle into the other rectangle. *puton* is defined as a function for a pair of symbolic representations. We show the conditions for success of *puton*. *embed* cannot be defined on the symbolic representation as *puton*, but it generates a solution for superposition that cannot be generated by *puton*. We explain these two operations, discuss their properties, and compare them. We also present an algorithm for superposing multiple rectangles using these two operations.

This paper is organized as follows. In Section II, we briefly explain qualitative spatial reasoning, which is the foundation of our approach. In Section III, we define the target object and describe its qualitative representation. In the following two sections, we describe the operations for superposing a pair of qualitative rectangles, and discuss our reasoning regarding superposition. In Section IV, we discuss the *puton* operation, and in Section V, we discuss the *embed* operation. In Section VI, we describe an algorithm for superposing multiple rectangles and show a behavior of an implemented system. Finally, in Section VII, we present our conclusions.

II. Qualitative Spatial Reasoning

Qualitative spatial reasoning (QSR) is a method for representing an objective spatial entity qualitatively, rather than using exact numeral data, and for reasoning about the properties that hold on these data [6], [7]. It extracts only the necessary aspects of the objective spatial data and represents them symbolically. For example, quantitative representation of Figure 2 is as follows: "There are two objects: one is a rectangle whose nodes at the bottom left are (1,1) and the length of the two edges are 3 and 5; the other is a circle whose center is (5,7) and radius is 2." However, the figure can also be qualitatively represented as follows: "There are two objects that have a common part." This is sufficient information for a discussion of the positional relationships

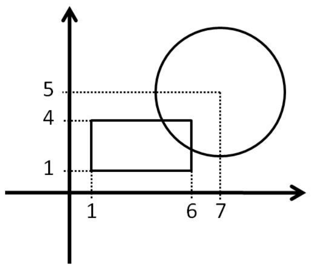

Figure 2. Qualitative versus quantitative representation

of objects. Moreover, if these objects are moving in time, their positional relationships changes. In some applications, we only need to focus on the instant in which disconnected objects change to become connected or in which connected objects change to have a relationship of inclusion, ignoring the exact distance between them or the exact size of their intersection part. QSR can provide a simple representation and reduce computational complexity.

Various QSR calculi or systems are available depending on what aspects of spatial data are interested, such as positional relationship, direction, distance, size, orientation or shape. Several studies have focused on qualitative spatial databases. For example, Wang and Liu developed a QSR application for a geospatial semantic web by constructing a qualitative spatial database that stores objects and their qualitative relations instead of coordinates, from the Geography Markup Language (GML) [11]. Santos and Amaral proposed an approach to develop a qualitative database by introducing qualitative identifiers such as direction and relative distance and applied it to data mining [12]. Although these studies have shown the effectiveness of qualitative spatial databases, further studies are required. Applications of QSR include geographic information systems, robot planning, navigation, and spatial databases, but few concrete applications have been developed to date.

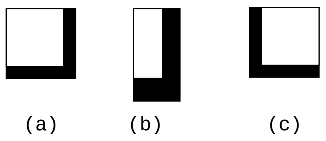

Figure 3. Qualitative rectangles

III. DESCRIPTION

First, we give a symbolic representation to the target objects.

We call a superposing entity *a unit*. A unit is a rectangle and is divided into WHITE, which should be visible, and BLACK, which should be hidden. BLACK is divided into a *core region* and a *non-core region*, which will be defined later. The outer side of a unit is called GRAY. The length of edges and the ratios of a unit and of each region are unfixed. In contrast, the orientation of a unit should be fixed. We only use rectangles situated in an upright position and do not consider those in an inclined orientation. This means that (a) and (b) in Figure 3 are regarded as equivalent, while (a) and (c) are regarded as different.

Each connected WHITE is called *a white region*, the core region and connected non-core regions are called *black regions*, and GRAY is called *a gray region*. White, black, and gray regions have attribute values related to visibility, denoted by 'w,' 'b,' and 'g.' 'w' and 'b' denote that regions should be visible and hidden, respectively. 'g' denotes that there is no requirement with respect to visibility.

Considering the structure of web page frames or the style of dividing a window into sub-windows used in many applications, we restricted the type of unit to those in Figure 6.

Any unit can be defined as a qualitative rectangle using the following operation that fits black plates into a white rectangle. Conversely, a qualitative rectangle obtained by this operation is only the units shown in Figure 6. Let $a * b$ represent a size of a unit whose length is a and height is b. Consider two black plates whose sizes are $x * b$ $(0 \leq x \leq a)$ and $a * y$ $(0 \leq y \leq b)$. Fit these plates into a white rectangle while preserving their orientation using either of the following procedures. Symbols enclosed in parentheses denote the names of unit types.

(0) No plate is fit (W).

(1) Only one of the plates is fit (B, I1, I2).

(2) Both plates are fit (L1, T1, PLUS).

(3) Extend L1 and T1, respectively, where the white region is added to the part on which the edge of size a or b is connected to the outer part (L2, T2).

Definition 1. *The unit obtained in this manner is said to be* valid*.*

The following theorem clearly holds.

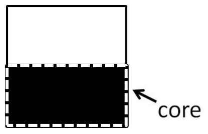

Figure 4. Core region and non-core region of straight-plate-unit

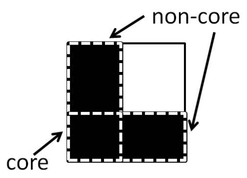

Figure 5. Core region and non-core region of cross-plates-unit

Theorem 2. *A unit is valid iff (i) the whole shape is rectangular, (ii) it has one connected* BLACK*, and (iii) all its white regions are convex.*

Types I1 and I2 are called *straight-plate-unit*s. Types L1, L2, T1, T2, and PLUS are called *cross-plates-unit*s.

For all units other than the W-type unit, the *core region* is defined. For B-type and straight-plate-units, the core region is defined as the entire BLACK (Figure 4). For cross-plates-units, the core region is defined as the intersection of the two plates, and the region not included in the core region is called the *non-core region* (Figure 5).

We denote the core region of a unit X by $Core_X$. The valid unit can be uniquely represented as a quadruple of attribute values composed of $Core_X$'s upper region, right region, lower region, and left region. We call this *a representation for a unit*. For example, the representation for the unit in Figure 5 is $\langle b, b, g, g \rangle$ because the core region has black regions in its upper side and right side, whereas it is connected to the outside in its lower side and left side. Note that the positional relationships of regions are preserved even if the size of a unit is changed.

Let V, R $(R \subset V^4)$, and T indicate a set of attribute values, a set of representations for units, and a set of types, that is:

$V = \{b, w, g\}$

$R = \{\langle r_1, r_2, r_3, r_4 \rangle \mid \text{a representation for a valid unit}\}$

$T = \{\text{'B','W','I1','I2','L1','L2','T1','T2','PLUS'}\}$

The function $rotate(r)$, which denotes a $\pi/2$ clockwise rotation of a unit r, and the function $symm(r)$, which denotes a symmetric transformation of a unit r, are defined as follows:

Let r be $\langle r_1, r_2, r_3, r_4 \rangle$.

$rotate : R \rightarrow R$

$rotate(\langle r_1, r_2, r_3, r_4 \rangle) = \langle r_2, r_3, r_4, r_1 \rangle$

$symm : R \rightarrow R$

$symm(\langle r_1, r_2, r_3, r_4 \rangle) = \langle r_1, r_4, r_3, r_2 \rangle$

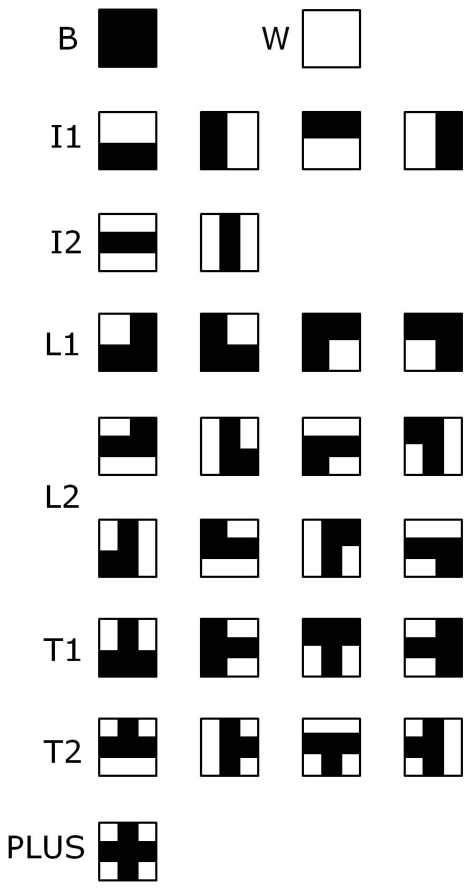

Figure 6. All valid units

The function *type* that defines the type for a representation $r \in R$ is defined as follows:

$type : R \rightarrow T$

$type(\langle g, g, g, g\rangle)$	= 'B'
$type(\langle w, w, w, w\rangle)$	= 'W'
$type(\langle w, g, g, g\rangle)$	= 'I1'
$type(\langle w, g, w, g\rangle)$	= 'I2'
$type(\langle b, g, g, b\rangle)$	= 'L1'
$type(\langle b, g, w, b\rangle)$	= 'L2'
$type(\langle b, b, g, b\rangle)$	= 'T1'
$type(\langle b, b, w, b\rangle)$	= 'T2'
$type(\langle b, b, b, b\rangle)$	= 'PLUS'

For representations $r, r' \in R$, if $r' = rotate(r)$ or $r' = symm(r)$ holds, then $type(r')$ is defined as $type(r)$.

Note that W-type is defined with the assumption that a tiny core region exists and is surrounded by white regions, as $Core_X$ does not exist.

The projections from $r \in R$ to its elements are defined as follows:

$up/dn/lt/rt : R \rightarrow V$

Let r be $\langle r_1, r_2, r_3, r_4\rangle$.

$up(r)$	$= r_1$
$rt(r)$	$= r_2$
$dn(r)$	$= r_3$
$lt(r)$	$= r_4$

IV. Reasoning about Superposition: puton

A. The principle

When n ($n \geq 3$) units are given, we consider the manner of their superposition in which all white regions are visible and all black regions are hidden.

Here, we place units sequentially. k-th unit ($n \geq k \geq 2$) should be placed on the figure composed of $k - 1$ units so that at least one region of the former is placed on at least one region of the latter. That is, we do not consider the placement in which, after two units are placed in a disconnected manner, a third unit is placed onto the black region of the two rectangles simultaneously. Thus, there should be at least one W-type unit. Here, we assume that there is one W-type unit. When more than one W-type unit exists, the scenario can be considered similarly. Then, the only one connected rectangular `BLACK` should be visible when superposition of $n - 1$ units is completed.

There are only two operations, *puton* and *embed*. *puton* is an operation of superposing the core regions of two valid units, whereas *embed* is an operation of superposing the whole unit onto a part of another unit. We describe these operations in detail.

B. Superposing the core regions

First, we describe *puton* operation.

Definition 3. *Suppose that a straight-plate-unit Y is put on a unit X. Let $Core_X$ and $Core_Y$ be the core regions of X and Y, respectively. The superposition in which $Core_Y$ is placed exactly on $Core_X$ is called* puton operation.

Let Z be the resultant figure of *puton*, and let $Core_Z$ be the superposed region of $Core_X$ and $Core_Y$. We extend a representation for a unit to be available as a representation for Z. *A representation for Z* is a quadruple of the attribute values of visible regions surrounding $Core_Z$.

First, we compute the attribute values of the regions around $Core_Z$. We define the function *on*, which computes the attribute value of the visible region when the second region is placed exactly on the first region, from the attribute values of the two regions.

$on : V \times V \rightarrow V \cup \{fail\}$

$on(b, b)$ =	b
$on(b, w)$ =	w
$on(w, w)$ =	$fail$
$on(w, b)$ =	$fail$
$on(g, v)$ =	v where $v \in V$
$on(v, g)$ =	v where $v \in V$

'$fail$' means that the operation failed in that case.

When the result is not $fail$, X's black regions are sometimes visible in Z. If they are connected with $Core_Z$

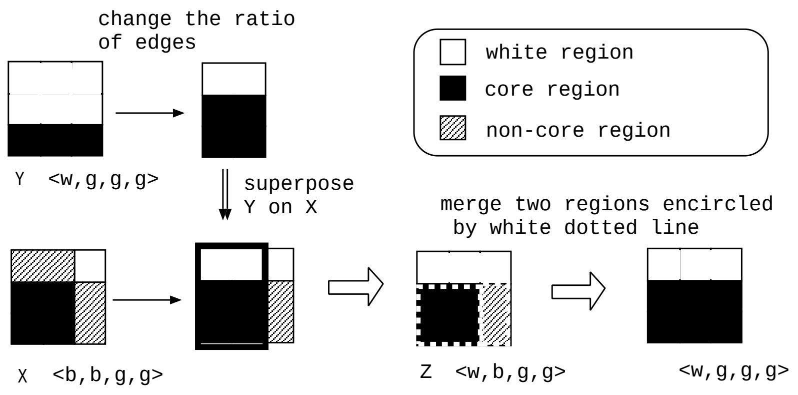

Figure 7. A case in which $merge$ is necessary

by lines, it is necessary to merge them to define the merged region as a new $Core_Z$. For example, in Figure 7, X and Y are represented as $\langle b, b, g, g\rangle$ and $\langle w, g, g, g\rangle$, respectively. When we place Y on X such that $Core_Y$ is placed on $Core_X$, the resultant figure Z is represented as $\langle w, b, g, g\rangle$. X's non-core region is visible and is connected with $Core_Z$ by a line. Then, this region is merged with $Core_Z$. This function $merge$ is defined as follows:

Let $r = \langle r_1, r_2, r_3, r_4\rangle$. If r satisfies $\bigwedge_{i=1,\dots,4}(r_i \neq fail)$, then $merge$ can be defined.

$merge : V^4 \to R$

$merge(r) =$

$$\begin{cases} \langle g, r_2, g, r_4\rangle & \text{if } r_1 = b \wedge r_2 \neq b \wedge r_3 = b \wedge r_4 \neq b \\ \langle r_1, g, r_3, g\rangle & \text{if } r_1 \neq b \wedge r_2 = b \wedge r_3 \neq b \wedge r_4 = b \\ \langle g, r_2, r_3, r_4\rangle & \text{if } r_1 = b \wedge r_2 \neq b \wedge r_3 \neq b \wedge r_4 \neq b \\ \langle r_1, g, r_3, r_4\rangle & \text{if } r_1 \neq b \wedge r_2 = b \wedge r_3 \neq b \wedge r_4 \neq b \\ \langle r_1, r_2, g, r_4\rangle & \text{if } r_1 \neq b \wedge r_2 \neq b \wedge r_3 = b \wedge r_4 \neq b \\ \langle r_1, r_2, r_3, g\rangle & \text{if } r_1 \neq b \wedge r_2 \neq b \wedge r_3 \neq b \wedge r_4 = b \\ \langle r_1, r_2, r_3, r_4\rangle & \text{otherwise} \end{cases}$$

Success of $puton$ operation

For valid units X and Y whose representations are $r = \langle r_1, r_2, r_3, r_4\rangle$ and $r' = \langle r'_1, r'_2, r'_3, r'_4\rangle$, respectively, the $puton$ operation that puts Y on X is defined as follows and succeeds if (c1) holds.

$puton : R \times R \to R$

$puton(r, r') =$

$merge(\langle on(r_1, r'_1), on(r_2, r'_2), on(r_3, r'_3), on(r_4, r'_4)\rangle)$

(c1) $\bigwedge_{i=1,\dots 4} on(r_i, r'_i) \neq fail.$

When the $puton$ operation succeeds, it results in a superposition in which no white region or black region is put on a white region, and the following property clearly holds due to the definition of $puton$.

Theorem 4. *If the puton operation succeeds,* BLACK *of the resultant figure is connected.*

When the $puton$ operation succeeds, it produces figures such as Figure 8. (a) is the result of putting I1-type unit on PLUS-type unit, that is, $puton(\langle b, b, b, b\rangle, \langle w, g, g, g\rangle)$. The result is $\langle w, b, b, b\rangle$. (b) is the result of putting I2-type unit on L1-type unit, that is, $puton(\langle b, g, g, b\rangle, \langle g, w, g, w\rangle)$. The result is $\langle g, w, g, w\rangle$. And (c) is the result of putting I1-type unit on L1-type unit, that is, $puton(\langle b, g, g, b\rangle, \langle g, g, g, w\rangle)$. The result is $\langle g, g, g, w\rangle$. When superposing multiple units, we superpose another rectangle on these figures. In this case, these figures should satisfy two more conditions for continue superposition: *effectiveness* and *validity*.

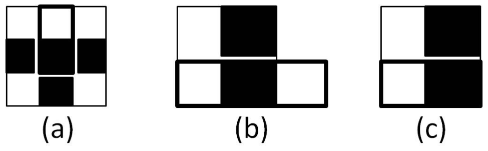

Figure 8. Resultant figures when $puton$ succeeds

Let Z be the resultant figure of superposing X and Y.

Definition 5. *Z has only one connected* BLACK *that is visible and rectangular, then Z is said to be* effective.

Definition 6. *If Z's entire shape is rectangular and all of its white regions are convex, then Z is said to be* valid.

From Theorem 4, if Z is valid, then Z is a valid unit.

When $n - 1$ units are superposed, the resultant figure should have only one connected visible BLACK, which is finally hidden by placing the W-type unit. This explains why the resultant figure of $puton$ should be effective. For example, Figure 8(a) is not effective. Moreover, the figure obtained as intermediate data in the process of superposing $n - 1$ units should be valid for the following continuous superposition. For example, Figure 8(b) is not valid. Figure 8(c) is both effective and valid. The conditions of effectiveness and validity can be checked using the following rules.

Effectiveness

Let r be a representation for Z. If r satisfies (c2), then

Z's BLACK is rectangular.

(c2) $\bigwedge_{i=1,\ldots,4}(r_i \neq b)$,

Validity

Let $r = \langle r_1, r_2, r_3, r_4\rangle$ $r' = \langle r'_1, r'_2, r'_3, r'_4\rangle$ and $r'' = \langle r''_1, r''_2, r''_3, r''_4\rangle$ be representations for units X, Y and Z, respectively. For the entire shape of Z to be rectangular, the white region of Y should not be placed on GRAY of X. Moreover, all of Z's white regions are convex. Therefore, if (c3) and (c4) are satisfied, then Z is valid. In the followings, r_i is regarded as r_{i-4} when $i \geq 5$.

(c3) If there exists i ($1 \leq i \leq 4$) such that $r_i = r'_{i+2} = g$ and that satifies one of the followings:
(i) $r_{i+1} = b \wedge r'_{i+1} \neq g \wedge r_{i+3} = g \wedge r'_{i+3} = g$
(ii) $r_{i+1} = g \wedge r'_{i+1} = g \wedge r_{i+3} = b \wedge r'_{i+3} \neq g$
(iii) $r_{i+1} = b \wedge r'_{i+1} \neq g \wedge r_{i+3} = b \wedge r'_{i+3} \neq g$
(iv) $r_{i+1} = g \wedge r'_{i+1} = g \wedge r_{i+3} = g \wedge r'_{i+3} = g$

(c4) No i ($1 \leq i \leq 4$) exists such that satisfies either of the followings.
(i) $r''_i = r''_{i+1} = b$
(ii) $r''_i = r''_{i+1} = r''_{i+2} = w$
(ii) $(r''_i \neq g) \wedge (r''_{i+2} \neq g) \wedge (r''_{i+1} = b)$

C. Result of superposition: puton

In Definition 3, we defined the *puton* operation for the superposition of a straight-plate-unit and a unit. In this subsection, we extend this operation to any pair of unit types. We also discuss the effectiveness and validity of the resulting figures when *puton* succeeds.

1) Superposition on B/W type: Assume that we superpose some unit on the B-type. The resultant figure is effective if and only if we superpose the straight-plate-unit, and it is valid for any type.

In contrast, it is impossible to place any unit on the W-type.

2) Superposition of straight-plate-units: Assume that we superpose the straight-plate-unit on the straight-plate-unit. The resultant figure is not always valid because its entire shape may not be a rectangle. The resultant figure is always effective.

3) Superposition of the straight-plate-unit on the cross-plates-unit: In this case, the resultant figure is not always valid and not always effective.

4) Superposition of the cross-plates-unit on any type: In this case, the resultant figure is always invald in case of putting on the straight-plate-unit, but sometimes valid in case of putting on cross-plates-unit. It is always ineffective. However, the *puton* operation succeeds for several cases.

D. Success of extended puton operation

Here, we show the conditions under which the *puton* operation succeeds for any pair of units. In general, when the *puton* operation is performed on X and Y, WHITE should

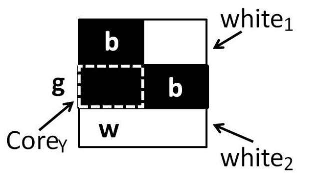

Figure 9. Representation of locations of white regions

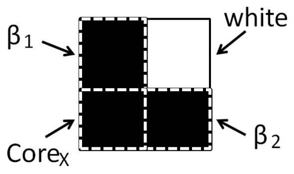

Figure 10. The regions to be hidden in L1-type

not be placed on X's white region. When Y is a cross-plates-unit, we have to consider its white region located in the inclined orientation from $Core_Y$. The location of the white region is represented as the occurrence either of b in adjacent elements or of b and w in adjacent elements in the representation for Y. For example, a representation for a unit in Figure 9 is $\langle b, b, w, g\rangle$. The sequence b, b represents the location of $white_1$, the upper left of $Core_Y$, and the sequence b, w represents that of $white_2$, the lower part of the unit. Therefore, the condition on WHITE can be represented as (c5).

(c5) Let $\langle r_1, r_2, r_3, r_4\rangle$ and $\langle r'_1, r'_2, r'_3, r'_4\rangle$ be representations of X and Y, respectively. There exists some i ($1 \leq i \leq 4$) such that $r_i = r'_{i+2} = g$, where r'_5 and r'_6 are regarded as r'_1 and r'_2, respectively.

Success of extended *puton* operation

For any pair of units X and Y, if (c1) and (c5) are satisfied, the *puton* operation succeeds.

The *puton* operation is an operation of superposing core regions. We can consider another operation in the manner in which specified parts of both units are superposed, for example, superposing non-core regions. However, no manner of superposition other than *puton* operation will yield an effective solution. We will prove this property.

Theorem 7. *When we superpose the straight-plate-unit on the cross-plates-unit, only the puton operation will yield an effective solution.*

Proof:

Consider the *puton* operation that places a straight-plate-unit Y on an L1-type unit X shown in Figure 10. In this case, BLACK is divided into three regions: one core region $Core_X$ and two non-core regions β_1 and β_2. Let $Core_Y$ be Y's core region.

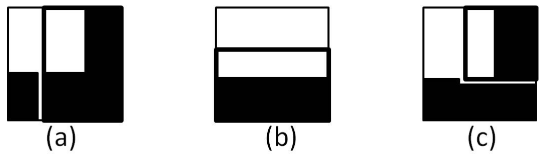

Figure 11. Three patterns of $embed$ operation

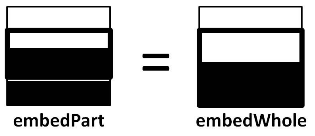

Figure 12. $embedWhole$ corresponding to $embedPart$

.

One or two of the $Core_X, \beta_1, \beta_2$ should be hidden so that the resultant superposed figure is effective.
(i) Only one region is hidden.

If only $Core_X$ is hidden, β_1 and β_2, which are disconnected, are visible. Therefore, the result is not effective. If only β_1 is hidden, $Core_X, \beta_2$ and $Core_Y$ are visible in the resultant figure. Considering the relative position of $Core_X, \beta_1$ and β_2, it is impossible to make a rectangle by merging $Core_X, \beta_2$ and $Core_Y$ and to hide β_1 at the same time. Therefore, the result is not effective. Similarly, the result is not effective if only β_2 is hidden.
(ii) Two regions are hidden.

Because β_1 and β_2 are disconnected, they are not simultaneously hidden by a single unit. If both $Core_X$ and β_1 are hidden, β_2 and $Core_Y$ are visible. We must place Y's regions onto both $Core_X$ and β_1 to hide them. Moreover, we must make a rectangle by merging β_2 and $Core_Y$. The only place where $Core_Y$ may be placed to satisfy both conditions is $Core_X$, and this placement is identical to the $puton$ operation.

According to the above analysis, the resultant figure is not effective by any operation other than the $puton$ operation.

Other cases can be similarly proved. □

V. Reasoning about Superposition: embed

A. Superposition by embedding

We can consider another superposition operation of $embed$. This operation embeds the whole of one unit into the whole or a part of `BLACK` in the other unit. It is defined on a pair of types, while the $puton$ operation is defined on a pair of representations for units.

Three patterns of embedding are possible, depending on the place of embedding.

1) Embed both into the core region and non-core region. For example, Figure 11(a) shows embedding of L1-type unit into L1-type unit.

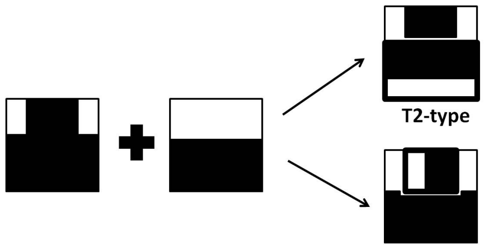

Figure 13. Solution differences between $embedWhole$ and $embedPart$

.

2) Embed only into the core region. For example, Figure 11(b) shows embedding of I1-type unit into I1-type unit. It is possible only when the background unit is straight-plate-unit.
3) Embed only into the non-core region. For example, Figure 11(c) shows embedding of I1-type unit into L1-type unit.

Remind the concept of a plate that is used in the construction of a valid unit. Two types of $embed$ operation can be defined using this plate.

Definition 8. *Placement of the whole unit in its entirety on a plate of the other unit is called an* $embedWhole$ *operation and placement on part of a plate of the other unit is called an* $embedPart$ *operation.*

The first case and the second case in the above patterns are $embedWhole$ operations, while the third case is an $embedPart$ operation.

B. Result of superposition: embed

Next, we discuss the result of $embed$ operation. The $embed$ operation always succeeds in the sense that any region is not put on `WHITE` of the background unit, and the entire shape of the resultant figure is a rectangle. Therefore, we discuss only the validity and effectiveness of the resultant figure.

1) Superposition on B/W type: Assume that we superpose some unit on the B-type. The resultant figure is effective if and only if we superpose the straight-plate-unit, and it is valid for any type.

In contrast, it is impossible to place any unit on the W-type.

2) Superposition of straight-plate-units: Assume that we superpose the straight-plate-unit on the straight-plate-unit. The resultant figure obtained by the $embed$ operation is not always valid because the white region may not be convex. The resultant figure is always effective.

Theorem 9. *If the result of* $embedPart$ *operation on a pair of straight-plate-units is valid and effective, then*

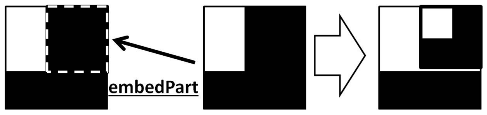

Figure 14. Ineffectiveness of $embedPart$ for a pair of cross-plates-units

an $embedWhole$ operation exists that generates the same result.

Proof.

Assume that the result of $embedPart$ operation on a pair of straight-plate-units is valid and effective, shown in Figure 12, for example. If we extend the BLACK of the foreground unit to fill the core region of the background unit, this corresponds to the $embedWhole$ operation, which generates the same result. □

It means that two figures in Figure 12 are regarded as qualitatively equivalent, and this is a characteristic of qualitative reasoning.

3) Superposition of the straight-plate-unit on the cross-plates-unit: In this case, the resultant figure obtained is not always valid and not always effective. $embedWhole$ and $embedPart$ may generate different solutions for the same pair. For example, if an I1-type unit is embedded into a T1-type unit, T2-type is generated by $embedWhole$, while T1-type is generated by $embedPart$ (Figure 13).

4) Superposition of the cross-plates-unit on any type: In this case, the resultant figure is always ineffective but can yield valid figures in some cases (See Table I).

Moreover, the following property holds.

Theorem 10. *If the result of embed operation on a pair of cross-plates-units is valid, then it is an $embedWhole$ operation.*

Proof.

Assume that the result of $embedPart$ is valid. In this case, BLACK portions of at least one plate of the background unit in the resulting figure are visible (Figure 14). These parts are the ones in which no unit is embedded. In contrast, BLACK of the foreground unit is visible and is not rectangular. The union of these portions of BLACK cannot make a shape of BLACK for any unit. Therefore, $embedPart$ never generates a valid solution. □

5) Valid solutions: Table I shows a pair of unit types where $embed$ operation generates a valid solution. In this table, rows show the unit in the foreground, and columns show the unit in the background. Only the solutions that differ from those generated by $puton$ operations are shown. U means there is no solution. The case of U* appears to be successful at first glance, but there is actually no solution. For example, Figure 15 shows the resultant figure obtained by the operation of $embed$ for L2 on T1. It is not valid because it is impossible to align line (1) and line (2).

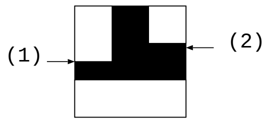

Figure 15. U*: Invalid example

Unlike the $puton$ operation, we cannot currently formalize rules for selecting the proper position for embedding or orientating units to obtain valid solutions. We can only say that generally, we place WHITE of a pair of superposing units on the adjacent position.

C. Comparison with puton operation

Table II compares validity and effectiveness between $puton$ and $embed$ operations. In the table, s and c indicate straight-plate-unit and cross-plates-unit, respectively.

Theorem 11. *(1) If there is a valid solution for the $puton$ operation, then there is a valid solution by the $embedWhole$ operation that generates the same solution.*
(2) Even if there is no valid solution for the $puton$ operation, there may be a valid solution by the $embedWhole$ operation.

Proof.

(1) We cannot obtain a valid solution in the case applying the $puton$ operation for cross-plates-unit on straight-plate-unit. Therefore, we consider the remaining three cases.

(i) superposing cross-plates-unit on cross-plates-unit

As both core regions are superposed, a plate of the foreground unit is put on a plate of the background unit. Let A be a visible part of a background unit and B be its invisible part. Moreover, let A' be a part of the foreground unit that is placed on the background unit and B' be its remaining part. Then, A' is a foreground of B by $puton$ operation. If we extend B so that it is a background of both A' and B', it is a solution of $embed$ operation. Since B corresponds to a single plate, its extention is qualitatively equivalent to the original one (Figure 16).

(ii) superposing straight-plate-unit on straight-plate-unit

This is proved in a similar way to the case (i).

(iii) superposing straight-plate-unit on cross-plates-unit

It is trivial due to the validity of the resultant figure.

(2) Only the solutions that differ from those generated by $puton$ operations are shown in Table I. □

$embed$ is an operation that is as essential as $puton$. When we superpose multiple units using both operations, we can sometimes obtain solutions that are not obtained only by a single operation. We can illustrate this using example of superposition of four units.

Consider superposition of four units X, Y, Z and W shown in Figure 17. The representations for X, Y, Z and W are $\langle b, b, g, b\rangle$, $\langle b, b, g, b\rangle$, $\langle w, g, w, g\rangle$, and $\langle w, w, w, w\rangle$,

fg\ bk	I1	I2	L1	L2	T1	T2	PLUS
I1	I1	I2	T1 L1 L2	T2 L2	T1 T2	T2	U*
I2	I2	I2	T1	T2	T1	T2	U
L1	L2	U	L1	L2 T2	T1	T2	U*
L2	L2	U	L2	L2	U*	U*	U
T1	T2	U	T1	T2	T1	T2	PLUS
T2	T2	U	U	U	T2	T2	U
PLUS	U	U	U	U	PLUS	U	PLUS

Table I
VALID SOLUTIONS FOR THE *embed* OPERATION

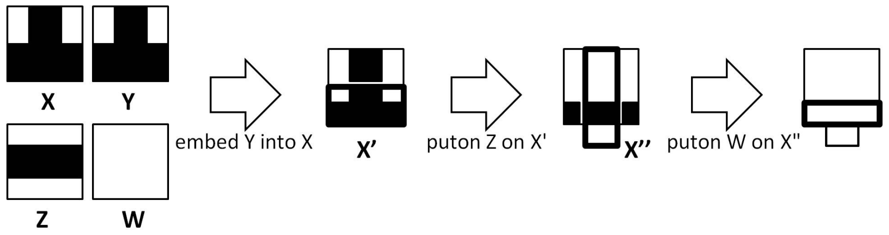

Figure 17. Solution by using *puton* and *embed* operations together

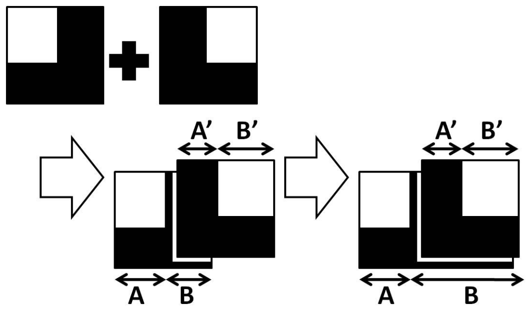

Figure 16. Superposing cross-plates-unit on cross-plates-unit

	puton		*embed*	
	validity	effectiveness	validity	effectiveness
s on s	some	always	some	always
s on c	some	some	some	some
c on c	some	never	some	never
c on s	never	never	some	never

Table II
COMPARISON BETWEEN *puton* AND *embed*

respectively. First, an effective solution for X, Y and Z should be generated. *puton* operation succeeds only for X and Z, that is, $puton(X, Z) = \langle w, g, w, g\rangle$, but the resultant figure is not valid. Therefore, we cannot continue the operation. In contrast, *embed* X into Y generates a valid solution X'. Next, put Z on this result X' using the *puton* operation. This yields an effective solution X''. Finally, by putting W on this result X'' using the *puton* operation we obtain the solution for superposing the four units.

```
Let $\Omega$ be a finite set of valid units that does not include
a W-type unit, where $|\Omega| \geq 2$, and $\omega$ is a W-type unit.

(1) Extract an arbitrary pair of $X$ and $Y$ from $\Omega$.
(2) If superposing $Y$ on $X$ generates a valid and
    effective solution,
        let $Z$ be the resulting figure.
    Otherwise, go back (1) to find another pair.
(3) If $|\Omega| = 2$,
        if $Z$ is effective, then $Z$ is a solution.
        else go back (1) to find another pair.
    else continue.
(4) Set $\Omega = \Omega - \{X, Y\} \cup \{Z\}$, and go to (1).

If a solution is generated, then the superposition of
$\Omega \cup \{w\}$ succeeds.
If it fails in all cases, then there is no solution.
```

Figure 18. Algorithm for superposing multiple units

VI. RECTANGLE REASONING SYSTEM

A. Algorithm for multiple unit superposition

We explain an algorithm for superposing multiple units. Here, superposition means either *puton* or *embed* operation. Selecting a unit from a given set of valid units, and perform superposition operation repeatedly to find a solution. The algorithm is shown in Figure 18.

B. Reasoning system

We implemented this algorithm using Prolog to code the main reasoning part and Java for the interface part.

We explain the behavior of the system.

1) Initial state
When the system is invoked, a basic frame is displayed that shows nine types of units (Figure 19).
2) Selecting the set of units
First, determine the first unit for superposing in the following manner: select the unit type by pushing the "select" button from "action" on the menu bar. Determine its orientation by pushing the "rotation" button. Repeat this procedure until n units are determined. Multiple units of the same type may be selected. The selected units are shown on the lower part of the frame (Figure 20).
3) Superposition
Next, judge if superposition succeeds and generate the solution if one is available. Find the superposing manner by pushing the "start" button from "file" on the menu bar. This opens a new window displaying the result. If superposition succeeds, the order of superposition and the positions of each unit are displayed (Figure 21). Otherwise, the window displays "No solution." If more than one solution is possible, only the first one found is shown.

VII. Conclusion and Future Work

A. Conclusions

We have discussed superposition of a pair of units and investigated the conditions that satisfy the result where all white regions are visible while all black regions are hidden in the resultant figure when visibility is specified by a user.

- A pair of straight-plate-units always produces an effective solution either by the *puton* operation or by the *embed* operation.
- The straight-plate-unit on cross-plates-unit can produce an effective solution in some cases either by the *puton* operation or by the *embed* operation. If a solution generated by the *puton* operation is valid, then it is also generated by the *embed* operation.
- The cross-plates-unit on any type can produce no effective solution.

As for the last case, we have shown which pairs can generate valid solutions.

We also presented an algorithm for superposing a set of units and implemented this system.

This work is the first study to focus on object placement with superposition and demonstrates a new application of QSR.

B. Future Works

We admit only units constructed using two specific plates. As a result, we have constraints both on BLACK and WHITE of a unit: BLACK should be one connected and all white regions are rectangles.

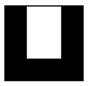

Figure 22. A unit constructed using three plates

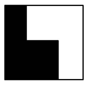

Figure 23. Allowing a non-rectangular white region

Assume that we admit a unit obtained by packing three plates (Figure 22). The above constraint on BLACK still exists. In this case, there are two core regions.

Next, consider that we weaken the constraint on WHITE. Assume that we can admit a white region that is not rectangular (Figure 23). In this case, a white region exists at the position over the black region viewed from the core region, whereas in the current definition, a white region can exist either on the adjacent or inclined orientation of the core region.

Currently, we can represent each unit uniquely by a representation using four directions of the core region. However, if we weaken the constraints and extend the target objects, we must change this representation, as the above consideration shows. This process is not straightforward, but we hope to weaken these constraints in future.

References

[1] G. Birgin, R. D. Lobato, and R. Morabito, "An effective recursive partitioning approach for the packing of identical rectangles in a rectangle," *Journal of the Operational Research Society,* vol. 61, pp. 306-320, 2010.

[2] A. S. Lapaugh, "Layout algorithm for VLSI design," *ACM Computing Surveys*, vol. 28, no. 1, pp. 59-61, 1996.

[3] H. Freeman, "Computer name placement," in *Geographical Information Systems 1*, D. J. Maguire, M. F. Goodchild, and D. W. Rhind, Eds. John Wiley, 1991, pp. 449-460.

[4] J. Li, C. Plaisant, and B. Shneriderman, "Data object and label placement for information abundant visualizations," in *Proceedings of the Workshop of New Paradigms Information Visualization and Manipulation (NPIV98)*, 1998, pp. 41-48.

[5] M. Aliello, I. E. Pratt-Hartmann, and J. F. A. K.Van Benthem, Eds., *Handbook of Spatial Logics*. Springer-Verlag, 2007.

[6] A. Cohn and S. Hazarika, "Qualitative spatial representation and reasoning: an overview," *Fundamental Informaticae*, vol. 46, no. 1, pp. 1-29, 2001.

[7] A. Cohn and J. Renz, "Qualitative spatial representation and reasoning," *Handbook of Knowledge Representation*, Chapt. 13, pp. 551-596, F. van Harmelen, V. Lifschitz, and B. Porter, Eds., Elsevier, 2008.

[8] M. Egenhofer and R. Franzosa, "On the equivalence of topological relations," *International Journal of Geographical Information Systems*, vol. 9, no. 2, pp. 133-152, 1995.

[9] S. Kumokawa and K. Takahashi, "Rectangle reasoning: a qualitative spatial reasoning with superposition," in *Proceedings of 23rd Florida Artificial Intelligence Research Society Conference (FLAIRS23)*, 2010, pp. 150-151.

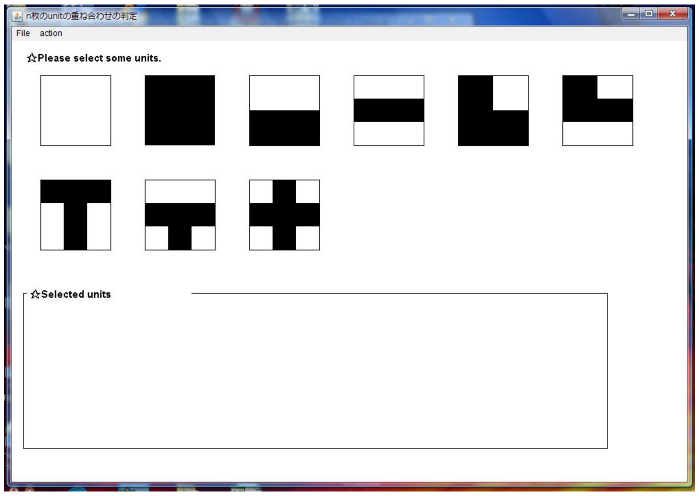

Figure 19. Screenshot of the system: 1. Initial state

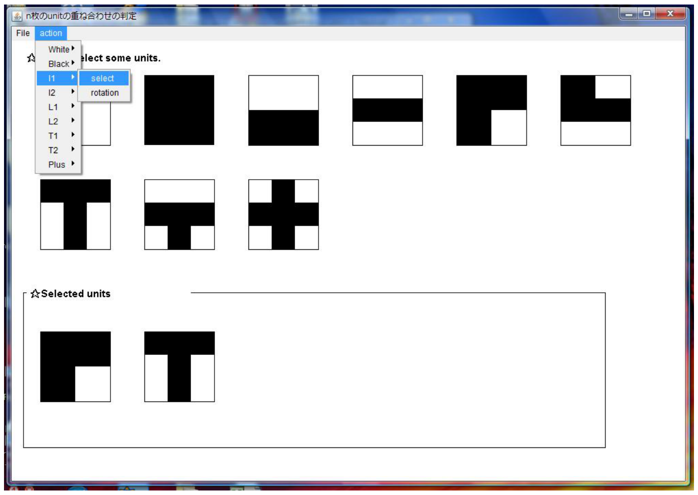

Figure 20. Screenshot of the system: 2. Selecting the set of units

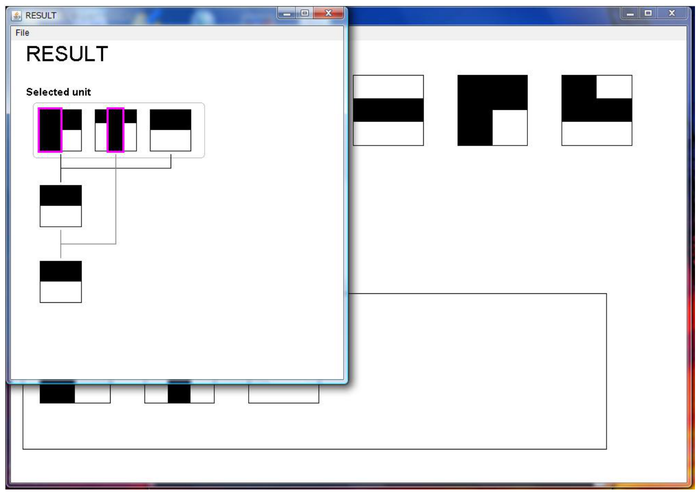

Figure 21. Screenshot of the system: 3. Superposition

[10] T. Konishi, and K. Takahashi, "Symbolic Representation and Reasoning for Rectangles with Superposition," The Third International Conference on Advances in Databases, Knowledge, and Data Applications (DBKDA 2011), pp. 71-76, January, 2011.

[11] S. Wang and D. Liu, "Qualitative spatial relation database for semantic web," in *First Asian Semantic Web Conference (ASWC)*, 2006, pp. 387-399.

[12] M. Santos and L. Amaral, "Geo-spatial data mining in the analysis of a demographic database," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 9, no. 5, pp. 374-384, 2005.