Group-based Generation of Referring Expressions

Funakoshi Kotaro * **Watanabe Satoru** † **Tokunaga Takenobu**

Department of Computer Science, Tokyo Institute of Technology
Tokyo Meguro Ôokayama 2-12-1, 152-8552, Japan
take@cl.cs.titech.ac.jp

Abstract

Past work of generating referring expressions mainly utilized attributes of objects and binary relations between objects in order to distinguish the target object from others. However, such an approach does not work well when there is no distinctive attribute among objects. To overcome this limitation, this paper proposes a novel generation method utilizing perceptual groups of objects and n-ary relations among them. The evaluation using 18 subjects showed that the proposed method could effectively generate proper referring expressions.

1 Introduction

In the last two decades, many researchers have studied the generation of referring expressions to enable computers to communicate with humans about objects in the world.

In order to refer to an intended object (the target) among others (distractors), most past work (Appelt, 1985; Dale and Haddock, 1991; Dale, 1992; Dale and Reiter, 1995; Heeman and Hirst, 1995; Horacek, 1997; Krahmer and Theune, 2002; van Deemter, 2002; Krahmer et al., 2003) utilized attributes of the target and binary relations between the target and distractors. Therefore, these methods cannot generate proper referring expressions in situations where there is no significant surface difference between the target and distractors, and no binary relation is useful to distinguish the target. Here, a *proper* referring expression means a concise and natural linguistic expression enabling hearers to identify the target.

For example, consider indicating object b to person P in the situation of Figure 1. Note that labels a, b and c are assigned for explanation to the readers, and person P does not share these labels with the speaker. Because object b is not distinguishable from objects a or c by means of their appearance, one would try to use a binary relation between object b and the table, i.e., "a ball to the right of the table". However, "to the right of" is not a discriminatory relation, for objects a and c are also located to the right of the table. Using a and c as a reference object instead of the table does not make sense, since a and c cannot be uniquely identified because of the same reason that b cannot be identified. Such situations have drawn less attention (Stone, 2000), but can frequently occur in some domains such as object arrangement (Tanaka et al., 2004).

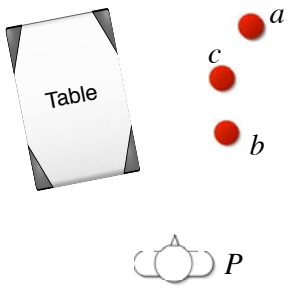

Figure 1: An example of problematic situations

In the situation of Figure 1, the speaker can indicate object b to person P with a simple expression "the front ball". In order to generate such an expression, one must be able to recognize the salient perceptual group of the objects and use the n-ary relative relations in the group.

*Currently at Honda Research Institute Japan Co., Ltd.
†Currently at Hitachi, Ltd.

Proceedings of the Fourth International Natural Language Generation Conference, pages 73–80,
Sydney, July 2006.

To overcome the problem described above, Funakoshi et al. (2004) proposed a method of generating Japanese referring expressions that utilizes n-ary relations among members of a group. They, however, dealt with the limited situations where only homogeneous objects are randomly arranged (see Figure 2). Thus, their method could handle only spatial n-ary relation, and could not handle attributes and binary relations between objects which have been the main concern of the past research.

In this paper, we extend the generation method proposed by (Funakoshi et al., 2004) so as to handle object attributes and binary relations between objects as well. In what follows, Section 2 shows an extension of the SOG representation that was proposed in (Funakoshi et al., 2004). Our new method will be described in Section 3 and evaluated in Section 4. Finally we conclude the paper in Section 5.

2 SOG representation

Funakoshi et al. (2004) proposed an intermediate representation between a referring expression and the situation that is referred to by the expression. The intermediate representation represents a course of narrowing down to the target as a sequence of groups from the group of all objects to the singleton group of the target object. Thus it is called SOG (Sequence Of Groups).

The following example shows an expression describing the target x in Figure 2 with the corresponding SOG representation below it. Since Japanese is a head-final language, the order of groups in the SOG representation can be retained in the linguistic expression.

hidari oku ni aru$_{(1)}$ *mittu no tama no uti no*$_{(2)}$
itiban migi no tama$_{(3)}$
(the rightmost ball$_{(3)}$ among the three balls$_{(2)}$
at the back left$_{(1)}$)

SOG:$[\{a, b, c, d, e, f, x\}, \{a, b, x\}, \{x\}]$,
where $\{a, b, c, d, e, f, x\}$ denotes all objects in the situation, $\{a, b, x\}$ denotes the three objects at the back left, and $\{x\}$ denotes the target.

2.1 Extended SOG

As mentioned above, (Funakoshi et al., 2004) supposed the limited situations where only homogeneous objects are randomly arranged, and considered only spatial subsumption relations between consecutive groups. Therefore, relations between groups are not explicitly denoted in the original SOGs as shown below.

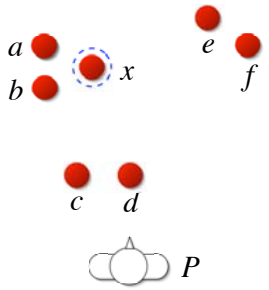

Figure 2: An example from (Funakoshi et al., 2004)

SOG: $[G_0, G_1, \ldots, G_n]$
G_i: a group

In this paper, however, other types of relations between groups are also considered. We propose an extended SOG representation where types of relations are explicitly denoted as shown below. In the rest of this paper, we will refer to this extended SOG representation by simply saying "SOG".

SOG: $[G_0 R_0 G_1 R_1 \ldots G_i R_i \ldots G_n]$
G_i: a group
R_i: a relation between G_i and G_{i+1}

2.2 Relations between groups

R_i, a relation between groups G_i and G_{i+1}, denotes a shift of attention from G_i to G_{i+1} with a certain focused *feature*. The feature can be an attribute of objects or a relation between objects. There are two types of relations between groups: *intra-group relation* and *inter-group relation*.

Intra-group relation When R_i is an intra-group relation, G_i subsumes G_{i+1}, that is, $G_i \supset G_{i+1}$. Intra-group relations are further classified into the following subcategories according to the feature used to narrow down G_i to G_{i+1}. We denote these subcategories with the following symbols.

$\xrightarrow{space}$: spatial subsumption
$\xrightarrow{type}$: the object type
$\xrightarrow{shape}$: the shape of objects
$\xrightarrow{color}$: the color of objects
$\xrightarrow{size}$: the size of objects

With respect to this classification, (Funakoshi et al., 2004) dealt with only the $\xrightarrow{space}$ relation.

Inter-group relation When R_i is an inter-group relation, G_i and G_{i+1} are mutually exclusive, that is, $G_i \cap G_{i+1} = \phi$. An inter-group relation is a spatial relation and denoted by symbol $\Rightarrow$.

Example R_i can be one of $\overset{space}{\longrightarrow}$, $\overset{type}{\longrightarrow}$, $\overset{shape}{\longrightarrow}$, $\overset{color}{\longrightarrow}$, $\overset{size}{\longrightarrow}$ and $\Rightarrow$. We show a referring expression indicating object $b1$ and the corresponding SOG in the situation of Figure 3. In the SOG, $\{all\}$ denotes the total set of objects in the situation. The indexed underlines denote correspondence between SOG and linguistic expressions. As shown in the figure, we allow objects being on the other objects.

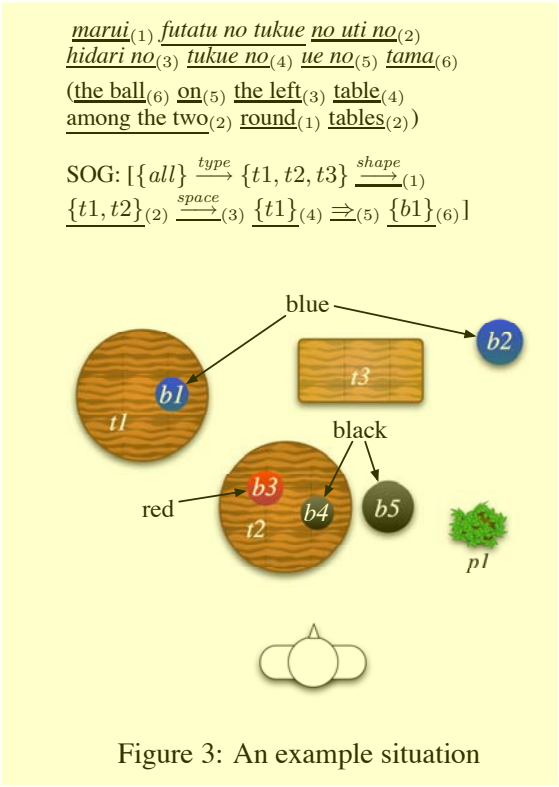

Figure 3: An example situation

3 Generation

Our generation algorithm proposed in this section consists of four steps: perceptual grouping, SOG generation, surface realization and scoring. In the rest of this section, we describe these four steps by using Figure 3 as an example.

3.1 Step 1: Perceptual grouping

Our algorithm starts with identifying groups of objects that are naturally recognized by humans. We adopt Thórisson's perceptual grouping algorithm (Thórisson, 1994) for this purpose. Perceptual grouping is performed with objects in the situation with respect to each of the following features: *type*, *shape*, *color*, *size*, and *proximity*. Three special features, *total*, *singleton*, and *closure* are respectively used to recognize the total set of objects, groups containing each single object, and objects bounded in perceptually significant regions (table tops in the domain of this paper). These three features are handled not by Thòrisson's algorithm but by individual procedures.

Type is the most dominant feature because humans rarely recognize objects of different types as a group. Thus, first we group objects with respect to types, and then group objects of the same type with respect to other features (except for *total*).

Although we adopt Thórisson's grouping algorithm, we use different grouping strategies from the original. Thórisson (1994) lists the following three combinations of features as possible strategies of perceptual grouping.

- *shape* and *proximity*
- *color* and *proximity*
- *size* and *proximity*

However, these strategies are inappropriate to generate referring expressions. For example, because two blue balls $b1$ and $b2$ in Figure 3 are too much distant from each other, Thórisson's algorithm cannot recognize the group consisting of $b1$ and $b2$ with the original strategies. However, the expression like "the left blue ball" can naturally refer to $b1$. When using such an expression, we assume an implicit group consisting of $b1$ and $b2$. Hence, we do not combine features but use them separately.

The results of perceptual grouping of the situation in Figure 3 are shown below. Relation labels are assigned to recognized groups with respect to features used in perceptual grouping. We define six labels: `all`, `type`, `shape`, `color`, `size`, and `space`. Features *singleton*, *proximity* and *closure* share the same label `space`. A group may have several labels.

feature	`label`	recognized groups
total	`all`	$\{t1, t2, t3, p1, b1, b2, b3, b4, b5\}$
singleton	`space`	$\{t1\}$, $\{t2\}$, $\{t3\}$, $\{p1\}$, $\{b1\}$, $\{b2\}$, $\{b3\}$, $\{b4\}$, $\{b5\}$
type	`type`	$\{t1, t2, t3\}$, $\{p1\}$, $\{b1, b2, b3, b4, b5\}$
shape	`shape`	$\{t1, t2\}$, $\{t3\}$
color	`color`	$\{b1, b2\}$, $\{b3\}$, $\{b4, b5\}$
size	`size`	$\{b1, b3, b4\}$, $\{b2, b5\}$
proximity	`space`	$\{t2, t3\}$, $\{b1, b3, b4, b5\}$, $\{b3, b4, b5\}$
closure	`space`	$\{b1\}$, $\{b3, b4\}$

```
Target     # target object
AllGroups  # all generated groups
SOGList    # list of generated SOGs

01:makeSOG()
02: SOG = []; # list of groups and symbols
03: All = getAll(); # total set
04: add(All, SOG); # add All to SOG
05: TypeList = getAllTypes(All);
    # list of all object types
06: TargetType = getType(Target);
    # type of the target
07: TargetSailency = saliency(TargetType);
    # saliency of the target type
08: for each Type in TypeList do
    # {Table, Plant, Ball}
09:  if saliency(Type) ≥
                  TargetSaliency then
    # saliency: Table > Plant > Ball
10:   Group = getTypeGroup(Type);
      # get the type group of Type
11:   extend(SOG, Group);
12:  end if
13: end for
14:return
```

Figure 4: Function makeSOG

3.2 Step 2: SOG generation

The next step is generating SOGs. This is so-called *content planning* in natural language generation. Figure 4, Figure 5 and Figure 6 show the algorithm of making SOGs.

Three variables Target, AllGroups, and SOGList defined in Figure 4 are global variables. Target holds the target object which the referring expression refers to. AllGroups holds the set of all groups recognized in Step 1. Given Target and AllGroups, function makeSOG enumerates possible SOGs in the depth-first manner, and stores them in SOGList.

makeSOG (Figure 4) makeSOG starts with a list (SOG) that contains the total set of objects in the domain. It chooses groups of objects that are more salient than or equal to the target object and calls function extend for each of the groups.

extend (Figure 5) Given an SOG and a group to be added to the SOG, function extend extends the SOG with the group for each label attached to the group. This extension is done by creating a copy of the given SOG and adding to its end an intra-group relation symbol defined in Section 2.2 corresponding to the given label and group. Finally it calls search with the copy.

search (Figure 6) This function takes an SOG as its argument. According to the last group in the SOG (LastGroup), it extends the SOG as described below.

```
01:extend(SOG, Group)
02: Labels = getLabels(Group);
03: for each Label in Labels do
04:  SOGcopy = copy(SOG);
05:  add(--Label-->, SOGcopy);
06:  add(Group, SOGcopy);
07:  search(SOGcopy);
08: end for
09:return
```

Figure 5: Function extend

1. If LastGroup is a singleton of the target object, append SOG to SOGList and return.
2. If LastGroup is a singleton of a non-target object, find the groups that contain the target object and satisfy the following three conditions: (a), (b) and (c).
   (a) All objects in the group locate in the same direction from the object of LastGroup (*the reference*). Possible directions are one of "back", "back right", "right", "front right", "front", "front left", "left", "left back" and "on". The direction is determined on the basis of coordinate values of the objects, and is assigned to the group for the use of surface realization.
   (b) There is no same type object located between the group and the reference.
   (c) The group is not a total set of a certain type of object.

   Then, for each of the groups, make a copy of the SOG, and concatenate "⇒" and the group to the copy, and call search recursively with the new SOG.
3. If LastGroup contains the target object together with other objects, let the intersection of LastGroup and each group in AllGroups be NewG, and copy the label from each group to NewG. If NewG contains the target object, call function extend unless Checked contains NewG.
4. If LastGroup contains only non-target objects, call function extend for each group (Group) in AllGroups which is subsumed by LastGroup.

Figure 7 shows the SOGs generated to refer to object $b1$ in Figure 3.

1. $[\{all\} \xrightarrow{type} \{t1, t2, t3\} \xrightarrow{space} \{t1\} \Rightarrow \{b1\}]$
2. $[\{all\} \xrightarrow{type} \{t1, t2, t3\} \xrightarrow{shape} \{t1, t2\} \xrightarrow{space} \{t1\} \Rightarrow \{b1\}]$
3. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{space} \{b1\}]$
4. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{color} \{b1, b2\} \xrightarrow{space} \{b1\}]$
5. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{color} \{b1, b2\} \xrightarrow{size} \{b1\}]$
6. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{size} \{b1, b4, b3\} \xrightarrow{space} \{b1\}]$
7. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{size} \{b1, b4, b3\} \xrightarrow{color} \{b1\}]$
8. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{space} \{b1, b3, b4, b5\} \xrightarrow{space} \{b1\}]$
9. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{space} \{b1, b3, b4, b5\} \xrightarrow{color} \{b1\}]$
10. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{space} \{b1, b3, b4, b5\} \xrightarrow{size} \{b1, b3, b4\} \xrightarrow{space} \{b1\}]$
11. $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{space} \{b1, b3, b4, b5\} \xrightarrow{size} \{b1, b3, b4\} \xrightarrow{color} \{b1\}]$

Figure 7: Generated SOGs from the situation in Figure 3

```
01:search(SOG)
02: LastGroup = getLastElement(SOG);
     # get the rightmost group in SOG
03: Card = getCardinality(LastGroup);
04: if Card == 1 then
05:  if containsTarget(LastGroup) then
     # check if LastGroup contains
     # the target
06:   add(SOG, SOGList);
07:  else
08:   GroupList =
         searchTargetGroups(LastGroup);
      # find groups containing the target
09:   for each Group in GroupList do
10:    SOGcopy = copy(SOG);
11:    add(⇒, SOGcopy);
12:    add(Group, SOGcopy);
13:    search(SOGcopy);
14:   end for
15:  end if
16: elsif containsTarget(LastGroup) then
17:   Checked = [ ];
18:   for each Group in AllGroups do
19:    NewG = Intersect(Group, LastGroup);
       # make intersection
20:    Labels = getLabels(Group);
21:    setLabels(Labels, NewG);
       # copy labels from Group to NewG
22:    if containsTarget(NewG) &
         !contains(Checked, NewG) then
23:     add(NewG, Checked);
24:     extend(SOG, Group);
25:    end if
26:   end for
27: else
28:  for each Group of AllGroups do
29:   if contains(LastGroup, Group) then
30:    extend(SOG, Group);
31:   end if
32:  end for
33: end if
34:return
```

Figure 6: Function `search`

3.3 Step 3: Surface realization

A referring expression is generated by deterministically assigning a linguistic expression to each element in an SOG according to Rule 1 and 2. As Japanese is a head-final language, simple concatenation of element expressions makes a well-formed noun phrase[1]. Rule 1 generates expressions for groups and Rule 2 does for relations. Each rule consists of several subrules which are applied in this order.

[Rule 1]: Realization of groups

Rule 1.1 *The total set* ($\{all\}$) *is not realized.*
(Funakoshi et al., 2004) collected referring expressions from human subjects through experiments and found that humans rarely mentioned the total set. According to their observation, we do not realize the total set.

Rule 1.2 *Realize the type name for a singleton.*
Type is realized as a noun and only for a singleton because the *type* feature is used first to narrow down the group, and the succeeding groups consist of the same type objects until reaching the singleton. When the singleton is not the last element of SOG, particle "*no*" is added.

Rule 1.3 *The total set of the same type objects is not realized.*
This is because the same reason as Rule 1.1.

Rule 1.4 *The group followed by the relation* $\xrightarrow{space}$ *is realized as "[cardinality] [type] no-uti (among)", e.g., "futatu-no (two) tukue (desk) no-uti (among)". The group followed by*

[1]Although different languages require different surface realization rules, we presume perceptual grouping and SOG generation (Step 1 and 2) are applicable to other languages as well.

the relation $\Rightarrow$ is realized as "[cardinality] [type] no".

When consecutive groups are connected by other than spatial relations ($\xrightarrow{space}$ and $\Rightarrow$), they can be realized as a sequence of relations ahead of the noun (*type* name). For example, expression "the red ball among big balls" can be simplified to "the big red ball".

Rule 1.5 *Other groups are not realized.*

[Rule 2]: Realization of relations

Rule 2.1 *Relation $\xrightarrow{type}$ is not realized.*

See Rule 1.2.

Rule 2.2 *Relations $\xrightarrow{shape}$, $\xrightarrow{color}$ and $\xrightarrow{size}$ are realized as the expressions corresponding to their attribute values. Spatial relations ($\xrightarrow{space}$ and $\Rightarrow$) are realized as follows, where $|G_i|$ denotes the cardinality of G_i.*

Intra-group relation ($G_i \xrightarrow{space} G_{i+1}$)

If $|G_i| = 2$ (i.e., $|G_{i+1}| = 1$), based on the geometric relations among objects, generate one of four directional expressions "{*migi, hidari, temae, oku*} *no* ({right, left, front, back})".

If $|G_i| \geq 3$ and $|G_{i+1}| = 1$, based on the geometric relations among objects, generate one of eight directional expressions "*itiban* {*migi, hidari, temae, oku, migi temae, hidari temae, migi oku, hidari oku*} *no* ({right, left, front, back, front right, front left, back right, back left}-most)" if applicable. If none of these expressions is applicable, generate expression "*mannaka no* (middle)" if applicable. Otherwise, generate one of four expressions "{*hidari, migi, temae, oku*} *kara* j*-banme no* (j-th from {left, right, front, back})".

If $|G_{i+1}| \geq 2$, based on the geometric relations among objects, generate one of eight directional expressions "{*migi, hidari, temae, oku, migi temae, hidari temae, migi oku, hidari oku*} *no* ({right, left, front, back, front right, front left, back right, back left})".

Inter-group relation ($G_i \Rightarrow G_{i+1}$)

$|G_i| = 1$ should hold because of `search` in Step 2. According to the direction assigned by `search`, generate one of nine expressions : "{*migi, hidari, temae, oku, migi temae, hidari temaen, migi oku, hidari oku, ue*} *no* ({right, left, front, back, front right, front left, back right, back left, on})".

Figure 8 shows the expressions generated from the first three SOGs shown in Figure 7. The numbers in the parentheses denote coindexes of fragments between the SOGs and the realized expressions.

3.4 Step 4: Scoring

We assign a score to each expression by taking into account the relations used in the expression, and the length of the expression.

First we assign a cost ranging over $[0, 1]$ to each relation in the given SOG. Costs of relations are decided as below. These costs conform to the priorities of features described in (Dale and Reiter, 1995).

$\xrightarrow{type}$	: No cost (to be neglected)
$\xrightarrow{shape}$	: 0.2
$\xrightarrow{color}$	: 0.4
$\xrightarrow{size}$	: big(est): 0.6, small(est): 0.8, middle: 1.0
$\xrightarrow{space}, \Rightarrow$	: Cost functions are defined according to the potential functions proposed in (Tokunaga et al., 2005). The cost for relation "on" is fixed to 0.

Then, the average cost of the relations is calculated to obtain the relation cost, C_{rel}. The cost of surface length (C_{len}) is calculated by

$$C_{len} = \frac{\text{length(expression)}}{\max_i \text{length(expression}_\text{i})},$$

where the length of an expression is measured by the number of characters.

Using these costs, the score of an expression is calculated by

$$score = \frac{1}{\alpha \times C_{rel} + (1 - \alpha) \times C_{len}}.$$

α was set to 0.5 in the following experiments.

4 Evaluation

4.1 Experiments

We conducted two experiments to evaluate expressions generated by the proposed method.

Both experiments used the same 18 subjects and the same 20 object arrangements which were generated automatically. For each arrangement, all factors (number of objects, positions of objects, attributes of objects, and the target object) were randomly decided in advance to conform to the following conditions: (1) the proposed method can generate more than five expressions for the given target and (2) more than two other objects exist which are the same type as the target.

1. SOG: $[\{all\} \xrightarrow{type} \{t1, t2, t3\} \xrightarrow{space}_{(1)} \underline{\{t1\}}_{(2)} \Rightarrow_{(3)} \underline{\{b1\}}_{(4)}]$
   itiban hidari no$_{(1)}$ *tukue no*$_{(2)}$ *ue no*$_{(3)}$ *tama*$_{(4)}$ (the ball$_{(4)}$ on$_{(3)}$ the leftmost$_{(1)}$ table$_{(2)}$)
2. SOG: $[\{all\} \xrightarrow{type} \{t1, t2, t3\} \xrightarrow{shape}_{(1)} \underline{\{t1, t2\}}_{(2)} \xrightarrow{space}_{(3)} \underline{\{t1\}}_{(4)} \Rightarrow_{(5)} \underline{\{b1\}}_{(6)}]$
   marui$_{(1)}$ *futatu no tukue no uti*$_{(2)}$ *hidari no*$_{(3)}$ *tukue no*$_{(4)}$ *ue no*$_{(5)}$ *tama*$_{(6)}$
   (the ball$_{(6)}$ on$_{(5)}$ the left$_{(3)}$ table$_{(4)}$ among$_{(2)}$ the round$_{(1)}$ two tables$_{(2)}$)
3. SOG: $[\{all\} \xrightarrow{type} \{b1, b2, b3, b4, b5\} \xrightarrow{space}_{(1)} \underline{\{b1\}}_{(2)}]$
   itiban hidari no$_{(1)}$ *tama*$_{(2)}$ (the leftmost$_{(1)}$ ball$_{(2)}$)

Figure 8: Realized expressions

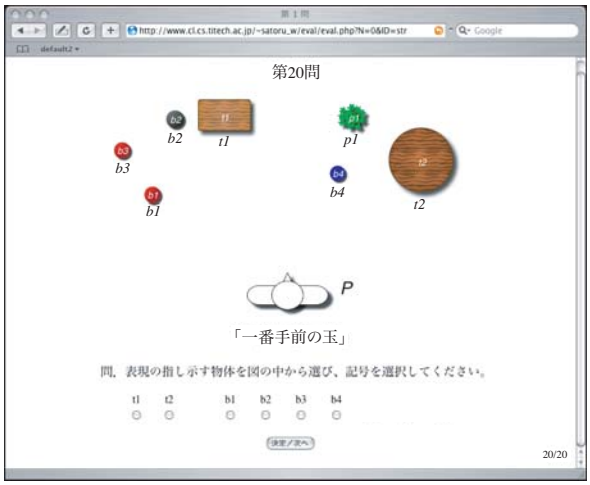

Figure 9: An example stimulus of Experiment 1

Figure 10: An example stimulus of Experiment 2

Experiment 1 Experiment 1 was designed to evaluate the ability of expressions to identify the targets. The subjects were presented an arrangement with a generated referring expression which gained the highest score at a time, and were instructed to choose the object referred to by the expression. Figure 9 is an example of visual stimuli used in Experiment 1. Each subject responded to all 20 arrangements.

Experiment 2 Experiment 2 was designed to evaluate validity of the scoring function described in Section 3.4. The subjects were presented an arrangement with a marked target together with the best five generated expressions referring to the target at a time. Then the subjects were asked to choose the best one from the five expressions. Figure 10 is an example of visual stimuli used in Experiment 2. Each subject responded to the all 20 arrangements. The expressions used in Experiment 2 include those used in Experiment 1.

4.2 Results

Table 1 shows the results of Experiment 1. The average accuracy of target identification is 95%. This shows a good performance of the generation algorithm proposed in this paper.

The expression generated for arrangement No. 20 (shown in Figure 9) resulted in the exceptionally poor accuracy. To refer to object $b1$, our algorithm generated expression "*itiban temae no tama* (the most front ball)" because $b1$ is the most close object to person P in terms of the vertical axis. Humans, however, chose the object that is the closest to P in terms of Euclidean distance. Some psychological investigation is necessary to build a more precise geometrical calculation model to solve this problem (Landragin et al., 2001).

Table 2 shows the results of Experiment 2. The first row shows the rank of expressions based on their score. The second row shows the count of human votes for the expression. The third row shows the ratio of the votes. The top two expressions occupy 72% of the total. This concludes that our scoring function works well.

5 Conclusion

This paper extended the SOG representation proposed in (Funakoshi et al., 2004) to generate refer-

Table 1: Result of Experiment 1

Arrangement No.	1	2	3	4	5	6	7	8	9	10
Accuracy	0.89	1.0	1.0	1.0	1.0	1.0	1.0	0.94	1.0	1.0

11	12	13	14	15	16	17	18	19	20	Ave.
1.0	0.94	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.17	0.95

Table 2: Result of Experiment 2

Rank	1	2	3	4	5	Total
Vote	134	125	59	22	20	360
Share	0.37	0.35	0.16	0.06	0.06	1

ring expressions in more general situations.

The proposed method was implemented and evaluated through two psychological experiments using 18 subjects. The experiments showed that generated expressions had enough discrimination ability and that the scoring function conforms to human preference well.

The proposed method would be able to handle other attributes and relations as far as they can be represented in terms of features as described in section 3. Corresponding surface realization rules might be added in that case.

In the implementation, we introduced rather *ad hoc* parameters, particularly in the scoring function. Although this worked well in our experiments, further psychological validation is indispensable.

This paper assumed a fixed reference frame is shared by all participants in a situation. However, when we apply our method to conversational agent systems, e.g., (Tanaka et al., 2004), reference frames change dynamically and they must be properly determined each time when generating referring expressions.

In this paper, we focused on two dimensional situations. To apply our method to three dimensional worlds, more investigation on human perception of spatial relations are required. We acknowledge that a simple application of the current method does not work well enough in three dimensional worlds.

References

Douglas E. Appelt. 1985. Planning English referring expressions. *Artificial Intelligence*, 26:1–33.

Robert Dale and Nicholas Haddock. 1991. Generating referring expressions involving relations. In *Proceedings of the Fifth Conference of the European Chapter of the Association for Computational Linguistics(EACL'91)*, pages 161–166.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.

Robert Dale. 1992. Generating referring expressions: Constructing descriptions in a domain of objects and processes. MIT Press, Cambridge.

Kotaro Funakoshi, Satoru Watanabe, Naoko Kuriyama, and Takenobu Tokunaga. 2004. Generating referring expressions using perceptual groups. In *Proceedings of the 3rd International Conference on Natural Language Generation: INLG04*, pages 51–60.

Peter Heeman and Graeme Hirst. 1995. Collaborating referring expressions. *Computational Linguistics*, 21(3):351–382.

Helmut Horacek. 1997. An algorithm for generating referential descriptions with flexible interfaces. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 206–213.

Emiel Krahmer and Mariët Theune. 2002. Efficient context-sensitive generation of descriptions. In Kees van Deemter and Rodger Kibble, editors, Information Sharing: Givenness and Newness in Language Processing. CSLI Publications, Stanford, California.

Emiel Krahmer, Sebastiaan van Erk, and André Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29(1):53–72.

Frédéric Landragin, Nadia Bellalem, and Laurent Romary. 2001. Visual salience and perceptual grouping in multimodal interactivity. In *Proceedings of International Workshop on Information Presentation and Natural Multimodal Dialogue (IPNMD)*, pages 151–155.

Matthew Stone. 2000. On identifying sets. In *Proceedings of the 1st International Conference on Natural Language Generation: INLG00*, pages 116–123.

Hozumi Tanaka, Takenobu Tokunaga, and Yusuke Shinyama. 2004. Animated agents capable of understanding natural language and performing actions. In Helmut Prendinger and Mituru Ishizuka, editors, *Life-Like Characters*, pages 429–444. Springer.

Kristinn R. Thórisson. 1994. Simulated perceptual grouping: An application to human-computer interaction. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, pages 876–881.

Takenobu Tokunaga, Tomofumi Koyama, and Suguru Saito. 2005. Meaning of Japanese spatial nouns. In *Proceedings of the Second ACL-SIGSEM Workshop on the Linguistic Dimentions of Prepositions and their Use in Computational Linguistics: Formalisms and Applications*, pages 93–100.

Kees van Deemter. 2002. Generating referring expressions: Boolean extensions of the incremental algorithm. *Computational Linguistics*, 28(1):37–52.