TransType: Development-Evaluation Cycles to Boost Translator's Productivity

Philippe Langlais (felipe@iro.umontreal.ca) and Guy Lapalme (lapalme@iro.umontreal.ca)
RALI/DIRO — Université de Montréal
C.P. 6128, succursale Centre-ville
H3C 3J7 Montréal, Canada
Phone: +1 (514) 343 6161
Fax: +1 (514) 343 2496

Marie Loranger (marie.loranger@statcan.ca)
119 Victor-Beaudry, Aylmer, Canada, J9H 7K1

Abstract. We present TransType: a new approach to Machine-Aided Translation in which the human translator maintains control of the translation process while being helped by real-time completions proposed by a statistical translation engine. The TransType approach is first presented through a series of prototypes that illustrate their underlying translation model and graphical interface. The results of two rounds of *in situ* evaluation of TransType prototypes are discussed followed by a set of lessons learned in these experiments. It will be shown that this approach is valued by translators but given the short time allotted for the evaluation, translators were not able to quantitatively increase their productivity. TransType is compared with other approaches and new perspectives are elaborated for a new version being developed in the context of a Fifth Framework European Community Project.

Keywords: Machine-Assisted Human Translation, Interactive Machine Translation, Target-Text Mediation, Word Completion, Statistical Translation Models, Statistical Language Models

1. Introduction

Translation needs are growing faster than machine translation (MT) technology improves. Therefore, there are more and more situations where MT is just not an acceptable solution, especially when high quality translation is required. Although it is hard to give precise figures about the rates of accuracy and level of fluency in current state of the art MT systems, trying an MT engine (for example, one that can be used on the web) to translate a real text can convince anyone of the limitations of the fully automatic approach. More scientific arguments against fully automatic MT may be found in (Kay, 1996).

Except for narrow sublanguage applications such as weather reports (Chandioux, 1989), automatic translations are more appropriate for gisting purposes and cannot attain publication quality without hu-

man revision. This is encouraging for (computational) linguists since people realize that more effort must still be invested in MT research. In the meantime, we believe that developing alternatives to fully automatic translation is a challenging but promising approach.

In our project, we take it for granted that high quality translation will continue to be done by professional translators fluent in both source and target languages. These people can produce accurate translations but are limited in the number of words they can translate in a certain period of time. Our goal is to augment their productivity by proposing context dependent *completions* during the translation process. Other tools, e.g. translation memories, online dictionaries and concordancers, can also help translators in their task, but in TransType we focus on speeding up translation input by reducing the number of keystrokes that must be entered to produce a translation. Of course, this is a surrogate measure for productivity, as will be shown by the evaluation results reported in this paper, but the approach is original enough to warrant investigation. Moreover the technology presented in this paper can eventually be integrated in the everyday work of a translator because it is embedded in a text editor which has now become the preferred way of producing a translation, compared to dictation which was more prevalent in the past.

In TransType (see Figure 1), a translation emerges from a series of alternating contributions by a human translator and the machine. The machine's contributions are basically proposals for parts of the target text, while the translator participates in various ways, including typing in pieces of target text and making corrections to a previous machine contribution. In all cases, the translator remains fully in control of the process: the machine must work within the constraints implicit in the user's contributions, and he or she is free to accept, modify, or ignore its proposals.

TransType, in which *the computer is helping the user*, can be contrasted with the "classical" Interactive Machine Translation (IMT) approach where the *user is helping the computer* by answering questions about word sense, ellipsis, phrasal attachments, etc. This latter vision was first implemented as part of the Kay's MIND system (Kay, 1973), and has been followed later on by others (Blanchon, 1991; Brown and Nirenburg, 1990; Maruyama and Watanabe, 1990; Whitelock et al., 1986). The evaluations reported for these systems focused on improving the question/answer process by having fewer questions or more user-friendly ones in order to produce a valid parse tree. No translation productivity measures were given partly because these systems were designed for monolingual users who do not have access to a professional translator.

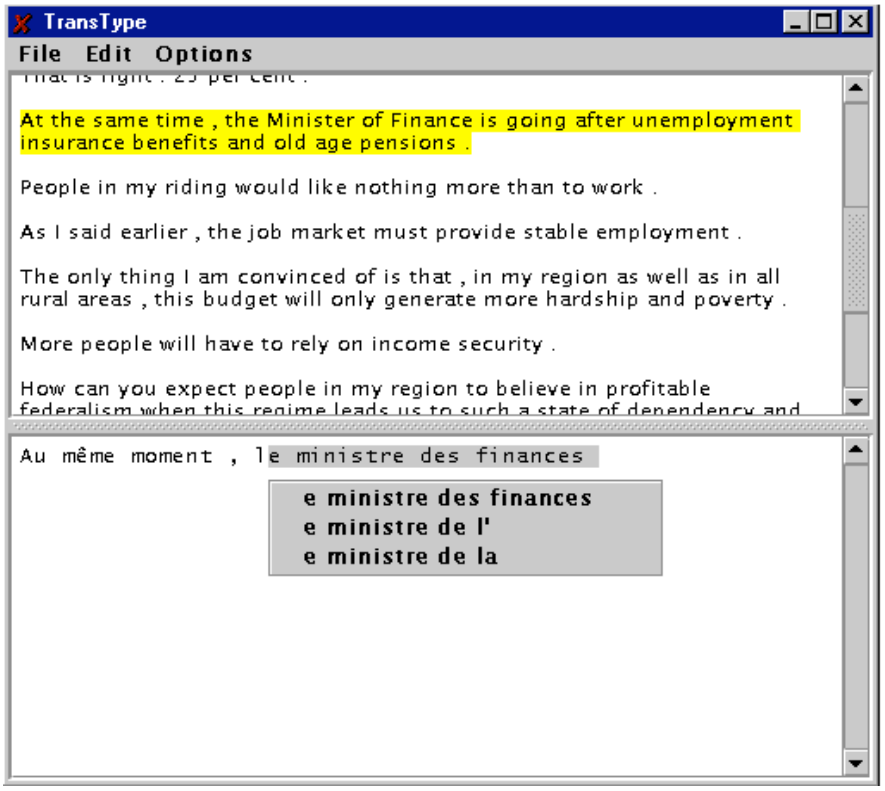

Figure 1. Example of an interaction in TRANSTYPE with the source text in the top half of the screen. The target text is typed in the bottom half with completions given by the menu at the insertion point. See `http://www-rali.iro.umontreal.ca/ttype-proto.en.html` for an animated screen dump of a short translation session. The screen dump is taken from the current Java interface in use for TT_2 but the set-up used in the experiments described in this paper was similar but with a TCL/Tk interface.

TRANSTYPE's interaction scenario has been called Target Text Mediated Interactive Machine Translation (TTM-IMT) because it shifts the focus of interaction from the *meaning* of the source text to the *form* of the target text. In this paper, we report on experiments that we conducted to turn the TTM-IMT concept into a real system a translator can work with. This effort comprised two development-evaluation cycles leading to the current prototype, which we now believe to be mature enough to be representative of what a useful TTM-IMT product could be. Among other things, this work required deciding on a number of ergonomic considerations. We organized two rounds of *in situ* evaluations whose results and assessments are the main contributions of this paper.

Although some drawbacks were identified, these user evaluations were very useful in demonstrating the interest of the innovative TransType concept in a real setting.

We will see that although the TransType translation engine performs very well at finding and proposing completions, translators do not always use them; and even when they do accept one, they may lose some time. So while TransType has yet to prove its efficiency, the tool is still appreciated by the users. This means that much work remains to be done on the user-interface aspect and we will point out research directions that are currently being pursued in order to reap the full potential of this innovative approach.

The next section provides details on the historical and scientific origins of our project and the different prototypes developed over the years. Section 3 describes the statistical models used in TransType and section 4 reports the experiments we performed to make TransType into a real system a translator may work with. Section 5 gives and analyses the results we have obtained and points out ways where TransType could be improved. In section 6, we compare TransType with related works. We conclude with some new perspectives for this approach.

2. Background

TransType originated from TransTalk(Brousseau et al., 1995) which was designed as a translation dictation system whose target text speech recognition accuracy was improved by taking into account the output of a statistical translation system appplied on the source text. For various technical and sociological reasons (one of them being that dictation is almost never used anymore by translators), TransTalk never developed past the demo stage.

The idea of using a combination of statistical language and translation models (described in section 2.1) for helping produce a translation was then adapted for translation typing. As shown in Table I, TransType evolved through a series of prototypes (TT_i), each one building on the experience gained from the previous one. Each new version benefitted from new insights but also from new tools and an increase in the computing power available on a desktop computer.

Foster et al. (1997) implemented TT_0, which proved that it was feasible for a computer system using a statistical translation model to produce real-time completions at the rate of good typist. Originally this was thought to imply producing about two new translations a second which seemed a daunting task. But with a slightly different design, TT_0 managed to propose in real-time the word which was most

Table I. Versions of the TRANSTYPE prototypes during the course of our project. This paper reports evaluation results for TT_{1a} and TT_{1b}

Name	Proof of concept TT_0	Experiments in this paper TT_{1a}	TT_{1b}	Next TT_2
Completions	word	word	word and fixed phrase	multiple words
User Interf.	line	Tcl/Tk GUI	Tcl/Tk GUI	Java GUI
Tracing	no	yes	yes	yes
Experiments		10 subjects 2 test cond.	9 subjects 3 test cond.	
References	(Foster et al., 1997)	(Langlais and Foster, 2000) (Langlais et al., 2001a)	(Langlais et al., 2000a) (Langlais et al., 2000b)	(Sema Spain et al., 2002)

.

likely to *complete* the ongoing translation. Under this simple scenario, a user could theoretically save about two thirds of the keystrokes needed to enter a given translation. The evaluation of this proof of concept system was done automatically by comparing the expected translation (as found in a translation corpus) and the output of the system. No user evaluation for TT_0 was conducted because only a rudimentary line-oriented interface had been developed for debugging and demo purposes. The user was presented completions one at a time and control keys were used to step through them and to select one. This prototype showed the feasibility of the underlying translation engine but was not really "usable" by translators in the context of their everyday work.

2.1. STATISTICAL ENGINE EMBEDDED WITHIN TRANSTYPE

The core of TRANSTYPE is a completion engine which comprises two main parts: a *generator* which produces a list of hypotheses that match the current (possibly null) prefix[1] and an *evaluator* which picks the best one according to the probabilistic score assigned to each completion hypothese. A diagram showing the interaction between the main components of the statistical engine of TRANSTYPE is provided in Figure 2.

[1] A prefix is defined technically as the first characters of a word typed by the user, not a linguistic or morphological prefix.

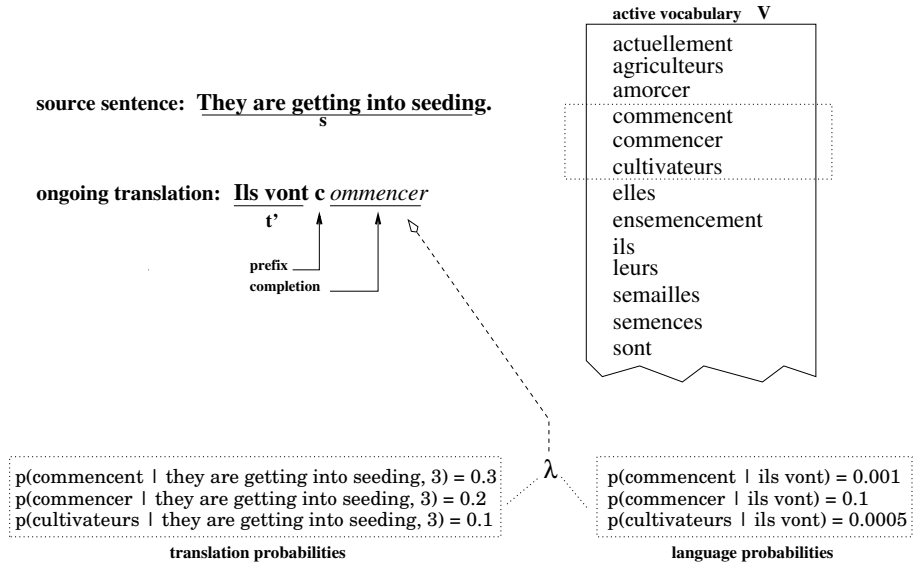

Figure 2. Components of the statistical engine of TRANSTYPE for a given source sentence. The user has already entered the prefix. TRANSTYPE chooses amongst the active vocabulary the word (or unit) yielding the best combined score given by the translation model and language model probabilities.

The evaluator is a function $p(t|t', s)$ which assigns to each target-text token t an estimate of its probability given a source text s (where s is the source sentence) and the tokens t' which precede t in the current translation of s. TT_0 uses a linear combination of separate predictions from a language model $p(t|t')$ and a translation model $p(t|s)$, where $\lambda = 0.6$ was chosen so as to optimize completion performance.

$$p(t|t', s) = \underbrace{p(t|t')\,\lambda}_{\text{language model}} + \underbrace{p(t|s)\,(1-\lambda)}_{\text{translation model}} \quad (1)$$

The language model is an interpolated trigram (Jelinek, 1990) trained on the Hansard corpus (about 50 million words), with 75% of the corpus used for relative-frequency parameter estimates, and 25% used to reestimate interpolation coefficients.

The translation model is a slight modification of an IBM model 2 (Brown et al., 1993) which takes into account invariant entities, such as English forms that almost invariably translate into French verbatim or that undergo a predictable transformation such as with numbers or dates. These forms are very frequent in the Hansard corpus.

3. Approach

3.1. TT_{1a}

Starting with TT_{1a}, TRANSTYPE took the form of a specialized text editor with an embedded Machine Translation engine as one of its components. We defined the following objectives for a better user-interface:

- hide the inner workings of the translation engine
- provide both the source text and appropriate completions for the target language translations
- embed the engine in a more convenient and intuitive text editor similar to the usual working environment of a translator.

3.1.1. *User interface elements*

We developed a first version of the editor in order to find the best way to display the text and the completions. We tried to display the text and its translation side by side but it seems that a synchronized display of the original text and its translation one above the other is preferable. We also tried displaying completions in a separate window, but we finally chose the set-up shown in Figure 1 where, at the most, seven[2] best completions are presented within a floating menu positioned at the cursor.

TRANSTYPE, like many other text editors, allows free movement of the cursor either with the mouse or the arrow keys. It also provides for the usual cutting and pasting of arbitrary selections of text. This requires a synchronization mechanism between the user interface and the translation engine of TRANSTYPE in order to follow these cursor movements and to update in real-time the context of the engine. Completions can either be stepped through using `PageUp` or `PageDown` keys (that is, the contents of the menu rotate); the top element of the menu is inserted in the text for easing of reading and can be accepted via the `Tab` key. A user can also directly click on any completion of the menu using the mouse. Table II indicates how many keystrokes are needed for accepting or rejecting a completion from the system.

This interface is implemented using a text widget in Tcl/Tk linked with a translation engine written in C++. The text widget is limited to editing plain character files. It is certainly not a full-featured text editor such as Microsoft Word, which allows for the formatting of characters

[2] (Miller, 1956) has shown that adults can memorize immediately about 7 elements of information.

Table II. Number of extra keystrokes accounted for accepting or rejecting a TransType completion

action	keys	# keys
accept the topmost completion	`tab`	1
	`return`	1
	click on element	1
select an alternative completion	`Page{Up|Down}` and accept	≥ 2
	click on alternative	1
reject a completion	intended letter	0
	`backspace`	1

using bold and italics, for paragraph indenting and centering and for creating figures and tables.

Given that our goal was simply to test the speed of typing the translation of isolated sentences, we did not need a full text processor but rather one that we could customize. More interestingly, having our own text editor allowed us to instrument it so that we could keep a log of all user actions and of their time of occurrence. This file was then analyzed off-line to compute measurements about the behavior of the users.

3.1.2. *Trace of a translation session*

To illustrate this process, we provide in Table III a one-sentence session. The first column indicates the best completion suggested by the system. The next two columns indicate respectively the characted typed by the user (+ indicates the acceptance key typed by the user). The source sentence to translate is *I shall return to this point in a few moments*, in which only one group of words is found in the lexicon[3] (*few moments*) with three likely translations (quelques minutes, quelques instants and quelques moments). Before the user types anything, TransType proposes the target word Je. Since this is what the user intended, she accepts this completion (which is indicated by a + in the second column).

The second token proves more problematic and clearly shows the weakness of mixing linearly the predictions of the language and the translation models. The machine's first proposed completion is le, which is not the word the user is looking for; thus she is forced to type its

[3] Lexicons and their role will be presented in the next section, but it is already instructive to follow the progress of a translation session

Table III. A one-sentence session illustrating the completion tasks.

Src.sent.:	*I shall return to this point in a few moments*
Tgt sent.:	Je reviendrai sur ce point dans quelques moments
In lexicon:	*few moments* → quelques minutes / quelques instants / quelques moments

Best completion	typed	resulting output	# keys	Target length
`Je`	+	Je␣	1	3
`le`	r	r	2	4
`etour`	e	e	3	5
`venir`	v	v	4	6
`iens`	i	i	5	7
`endrai`	+	endrai␣	6	14
`sur`	+	sur␣	7	18
`ce`	+	ce␣	8	21
`point`	+	point␣	9	27
`de`	d	d	10	28
`ans`	+	ans␣	11	32
`le`	q	q	12	33
`uelques instants`	u	u	13	34
`elques minutes`	e	e	14	35
`lques moments`	+	lques moments␣	15	49

first letter. TRANSTYPE adjusts to the user's input by proposing in turn several forms of the word retour (return). Note that TRANSTYPE came up with the completion retour on the strong recommendation of the translation model, while the language model considered it unlikely. Such situations, called *sequence breaks*, are responsible for many keystrokes needed to type a translation. Foster (2000) has developed an attractive alternative model which avoids many of these sequence breaks. The idea is basically to combine language and translation models by multiplication instead of addition as in TT_1. Adding model scores is equivalent to an *OR filter*, letting through predictions that are motivated by either the source sentence or the target sentence prefix, but not necessarily both; multiplying scores acts like an *AND filter*, letting through only those predictions that make sense from both source and target perspectives.

We can evaluate TRANSTYPE automatically by counting the number of keystrokes saved over a full translation session, assuming an "ideal"

user[4] who accepts the best completions as soon as they become available, does not change his mind once something has been typed, does not move the cursor with the mouse and does not erase whole words or part of words in the text. In the example of Table III, the target text contains 49 characters (42 letters and 7 spaces), the user has typed 15 keys to produce them (8 characters plus 7 approvals), so the percentage of keystrokes saved is 69%.

3.1.3. *Statistical engine modifications*
In TT_{1a}, we investigated several ways to partition the (t', s)-space of equation 1 by replacing λ by context-dependent interpolation coefficients $\lambda(\Theta(t', s)) \in [0, 1]$, which maps (t', s) into a set of equivalence classes. Intuitively, $\lambda(\Theta(t', s))$ should be high when s is more informative than t' and low otherwise. Unfortunately, this did not significantly improve the performance over a fixed linear combination scheme (Langlais and Foster, 2000). The only special case we consider when combining both models is at the very beginning of a sentence where we favor the translation model (by giving it a higher weight) against the language model. This does not have a strong impact on performance, however, although it prevents TransType from always initiating the translation of a sentence with the most frequent words in the training corpus.

3.2. TT_{1b}

The only user interface modification implemented between TT_{1a} and TT_{1b} was to allow the `Return` key to be used for accepting completions (instead of terminating the current sentence). This seemingly insignificant user-interface detail was slowing people in TT_{1a}, probably because `Return` is often used in other user interfaces for approval of a computer generated default choice.

From informal discussions with translators, we concluded that an important part of the translation process relies on lexicons. Actually, one of a translator's first tasks is often terminological research; and many translation companies employ specialized terminologists. The need for specialized lexicons becomes even more crucial in a machine translation application. Beyond the infrequent cases where, in a given thematic context, a word is likely to have a clearly preferred translation (*e.g.* bill/*facture vs* bill/*projet de loi*), lexicons are often the only means for a user to customize machine translation systems.

[4] One who would obtain the best productivity score under this experimental set-up

Table IV. Example of a user lexicon. This one has been automatically acquired from an excerpt of the Hansard corpus.

source	target
export enhancement program	programme de stimulation des exportations
	programme d'amélioration des exportations
farm debt review boards	bureaux d'examen de l'endettement agricole
federal cuts	compressions budgétaires fédérales
senior officials	hauts fonctionnaires
shipbuilding	construction navale

As TRANSTYPE is deeply user-oriented, we felt it was a desirable extension to the system to let users introduce specialized lexicons. This extension can be seen as a first step toward an adaptative version of TRANSTYPE, which is a very challenging issue.

TT_{1b} allows for the completion of both words and sequences of words (*units* hereafter). For instance, in the snapshot of Figure 1, we see that the first completion that TRANSTYPE proposes is the unit le ministère de finances (*the Minister of Finance*) as the most likely way of completing the current target material. In TT_{1b}, the units that may appear in the pop-up menu come from what we call a *user lexicon*. As it is user-dependent, we do not provide a formal definition, but Table IV provides some examples of its content: basically any source-target association between sequences of words. The automatic derivation of such lexicons has been fully described in Langlais et al. (2000a), but for our user evaluation, we compiled the user lexicon by hand because we were using a text from a small domain (baby feeding) for which we could not build terminology automatically. Moreover we wanted to be sure that the lexicon we used for the experiment was good enough to be useful for the translators.

Details of the implementation of the integration are described in Langlais *et al* (2001a). Basically, due to the simplicity of the decoder, a heuristic is used to decide whether the prediction of a target unit coming from a user lexicon is pertinent at a given time or not. If we could know, at completion time, which portion of the source text is currently being translated by the user, and furthermore that this portion was part of a source unit found in the lexicon, then a target association of this unit could be safely proposed. Unfortunately, we do not have this information, but we do know the contribution of each source word of the sentence being translated ($s_i, 1 \leq i \leq n$) to the

prediction of a given target word (t_j) at the target position j[5]. In the case of our IBM2-like model, this is given by equation 2, where $t(t_j|s_i)$ is a transfer probability (the probability that t_j is the translation of s_i) and $a(i|j,n)$ is an alignment probability (the probability that a source word in position i in a source sentence of n words is generating a target word in position j).

$$p(t_j|s_1^n) = \sum_{i=0}^{n} t(t_j|s_i)a(i|j,n) \tag{2}$$

Therefore, if there is one source word s_b which dominates this sum[6] and if s_b belongs to a source entry u_s in a lexicon, then any translation of u_s in our lexicon which starts with word t_j, may be predicted. The score of the unit prediction t_j^k is approximated quite crudely but it seems to apply well in our case where the associations in a user lexicon are well-formed, therefore potentially well scored by the language model:

$$\begin{array}{l} p(t_j^i|s_1^n) \approx p(t_j|s_1^n) + \delta \\ p(t_j^i|t') \approx p(t_j|t') \end{array} \tag{3}$$

where δ is any small positive value which favors a unit against its first word when this heuristic is applied[7].

3.3. TT_2

The TransType2 project envisages a substantially more ambitious research prototype by its completion date in March 2005. Fundamental enhancements are planned for both the user interface and the translation engine. The final prototype will be capable of running under both the Linux and Windows platforms in order to facilitate development as well as testing in a realistic user environment.

3.3.1. *Interface*

The TT_2 interface aims at providing a tool that users can realistically employ within their normal working environment, while at the same time supporting the research goal of testing the effectiveness of a variety of different interface and engine configurations. It will have the same

[5] s_i is the i_{th} word in s and s_0 is a special word to account from so-called *spurious words*

[6] In practice this fact is controlled via a threshold and other heuristics to account for many cases where two (or more) source words dominate the sum.

[7] By adding this δ value to the word prediction made by our translation model, we do not respect the properties a strict probabilistic engine; but given the simplicity of our decoder, which does not depend on previous predictions, this is not a problem.

basic layout as previous versions and will also be capable of recording and timestamping user actions.

To give the user more flexibility, the new interface will have a full range of editing functions available for the *source* text, including the ability to modify the default sentence segmentation. As with previous versions, translation in TT_2 will be on a sentence-by-sentence basis. For convenience, translation projects may be saved for later revision or completion; user preferences will also be storable. An important feature is the ability for users to define and *dynamically modify* their own lexicons of preferred term translations.

Completions in TT_2 will be highly configurable by the user. This includes the conditions under which completions are made—after every user action, only after the user has been quiet for a specified period, or only on demand; the length of completions, with minimum length directly settable and and maximum length controlled by a confidence threshold; and the maximum number of alternatives that should be displayed. The methods for accepting completions will also be more flexible than in TT_1, with the possibility to accept all or only part of a completion using the mouse, special keys, or a spoken command.

To allow for a modern, portable, and feature-rich interface, the language of implementation for the TT_2 interface will be Java.

3.3.2. *Engine*

The engine for TT_2 will be a true translation engine capable of making realistic completions for text units that range from a single word to the remainder of the current target sentence. The generate-and-evaluate paradigm used in TT_1 for next-word (or next-lexicalized-expression) completion will be replaced by a multi-word search capability typical of statistical MT. More powerful translation models, based on statistical and finite-state techniques, will be used. One of the main research challenges will be to make sentence-length completions with these models in real time.

Other novel features of the TT_2 engine will include the ability to adapt to different domains and to recently-observed text, and to dynamically incorporate explicit user input in the form of preferred term translations. The engine will also subsume the role of a traditional translation memory, in that it will be capable of detecting when the current source sentence has previously occurred in its training corpus. When this happens, the previous translation will be presented as a completion to the user, with a flag to indicate its status.

4. Evaluation

In Spring 2000, we performed an *in situ* evaluation of a version of TT_{1a} using the experimental protocol presented in section 4.1. Ten native French speaking translators visited our laboratory and used TransType for more than one hour each. Two of them were translation professors, two were translation students and the others were professional translators. The results of this study have been fully described in Langlais et al. (2000b; 2001b). Drawing conclusions from such a small sample of users can be quite risky but it must borne in mind that it was already an achievement for us to recruit many otherwise busy translators and have them spend two (unpaid) hours learning and using our prototype. Translators were also anxious about the fact that we would be evaluating the quality of their translation. We thus promised that we would only evaluate their typing speed and not the accuracy or fluency of their translation. A cursory look at the translations they produced shows that quality does not seem to have been a problem: they all produced reasonably clear and accurate translations of the source text.

Some interesting observations emerged from this evaluation. Only one translator actually managed to translate faster using TransType; this suggests that even in a very simple scenario, TTM-IMT translation is at least viable. Lack of training time is probably one reason for this otherwise disappointing result. The fact that real users do not systematically watch the screen when typing may also account for part of the problem.

A qualitative survey revealed that most users (actually nine out of ten) liked TransType and would be eager to try it in their daily work. However, they expressed the desire for a version of the system which would be able to propose completions beyond the single word level.

In the summer of 2001, we conducted another round of evaluations with TT_{1b} that took advantage of the lessons learned in the previous round. Nine translators visited our laboratory with the same mix of professors, students and professional translators as for TT_{1a}. Two translators participated in both evaluations but we did not compare their results given the same sample size. The goal of this evaluation was twofold: to measure if TransType saves its users time and to gauge if it helps in other ways, such as providing ideas for the translation of terms for which there is some hesitation. As the completions of TransType are correctly spelled, their selection reduces the number of misspellings in the target text. This is particularly useful for completed proper nouns and numbers which must always be carefully transcribed and are often error prone.

4.1. User Protocol

We asked translators with various work experience and expertise to use TransType in a controlled setting. We took for granted that their translations would be acceptable because we wanted to evaluate our system and not the translators themselves. All translators were given the same sentences to translate. In the first two steps, the source sentences were drawn from a Hansard file which was not part of our training corpus.

The protocol consisted of three steps:

1. **5 to 8 minutes without TransType**[8] to reassure the translators that our text editor was quite conventional: the usual keys for deletion, cursor movement, cutting and pasting are present. However there is no provision for formatting. This step measures the "natural" typing speed of each translator, i.e. their speed of thinking and typing a translation in our text editor but without TransType activated.

2. **15 to 20 minutes with TransType** in which the user types a translation while being able to select completions proposed by the system.

3. **5 to 8 minutes with a special lexicon** where the translators had to work with an excerpt of "Nutrition for Healthy Term Infants", a file downloaded from the Health Canada web site. We chose this text because it contained a relatively high number of specialized terms that we could add to a lexicon. Examining both the English and French texts, we handcrafted a lexicon which was then integrated into TransType using the technique described in section 3.2.

At the end of the experiment, users were asked to complete a short questionnaire and verbally describe their impressions on their experience with TransType. We also recorded whether or not the users looked at the keyboard while typing. This would suggest that they were missing some completions provided by TransType.

[8] In order to maintain a friendly atmosphere an exact timing of each translator was not kept; we just let the translators work until they had finished a complete sentence.

5. Results

5.1. Quantitative analysis

A theoretical evaluation of the translation engine (Foster et al., 1997) has shown that TransType can save about two thirds of the keystrokes needed to type a given translation (at least in situations where the text to be translated is close enough to those used for training). Unfortunately, the results given in table V are quite different. Users typed 69% of the required keystrokes because they did not take advantage of all the completions that were available to them. From the study of the logs which kept track of the users' actions, we inferred that users could have saved up to 68% of the characters if they had always chosen the best completions that were available to them.

These results are noticeably different from those of the TT_{1a} evaluation where users had typed 55% of the characters (see the last line of table V).

Table V. Number of characters typed, accepted by validating the completions of TransType, and erased. The fifth column reports the number of characters present in the text produced at Step 2 of our protocol. The last column shows the proportion of characters typed manually divided by the number of characters in the final text. The last two lines indicate the mean for TT_{1a} and TT_{1b} evaluation.

subject	# typed	# accepted	# erased	# final	% typed
1	1528	1001	181	2348	60%
2	791	411	181	972	66%
3	1234	582	149	1667	68%
4	1255	164	60	1360	88%
5	554	311	77	789	64%
6	1220	757	191	1786	62%
7	374	260	94	537	59%
8	634	352	178	809	64%
9	2198	332	407	2123	87%
TT_{1b}	1088	463	169	1377	69%
TT_{1a}	486	972	111	1347	55%

When a user accepted a completion, 39% of the time it was via the return key because the completion was the first choice, 38% with the return key after stepping through an average of 3.1 items in the list of

completions, and 23% of the time by selecting the completion with the mouse.

5.2. PRODUCTIVITY

We define productivity as the ratio of the number of characters in the final text over the time it took to produce the text. In the following, we express this quantity in terms of the number of characters produced in a minute. In interviews, almost all translators revealed that they thought that TRANSTYPE had improved their productivity. Unfortunately Table VI does not corroborate this favorable impression, because on the average, raw productivity went down by 17% – an improvement over TT_{1a}'s decline of 35%! Subject 2, who was especially slow in the first step, made a very productive use of completions. With the lexicon, users lost still more productivity. We will come back to this point later.

Table VI. The productivity of the translators at each step of the protocol described in section 4.1. Gain figures are computed with the productivity measured in step 1. The last line indicates the mean for all translators. The last two columns give the productivity and gain for step 2 but not counting the first 10 minutes in order to take into account the learning time involved in the interaction with TRANSTYPE.

Subject	Step 1	Step 2	Gain	Step 3	Gain	Step 2'	Gain
1	153	112	-27 %	89	-41 %	109	-29%
2	28	53	88 %	43	53 %	48	70%
3	114	94	-17 %	63	-45 %	79	-31%
4	86	84	-3 %	79	-9 %	89	3%
5	79	48	-39 %	57	-32 %	71	-10%
6	113	104	-7 %	58	-45 %	116	3%
7	53	37	-30 %	61	-7 %	52	-2%
8	64	40	-38 %	43	-32 %	38	-40%
9	145	119	-18 %	137	-17 %	152	5%
TT_{1b}	93	77	-17 %	70	-25 %	84	-10%
TT_{1a}	102	65	-35 %				

Questions can be raised about the learning curve for such an "unusual" tool as TRANSTYPE. We did some statistics by ignoring the first half (ten minutes) of Step 2. Of course, statistics are then computed on a much shorter time (8 to 10 minutes) but we can observe an important difference (10% loss instead of 17%). This saving does not seem to

depend on the speed of the user. The proportion of manually entered characters is still the same at about 70%.

These results are only preliminary and, as the participants told us themselves, the use of TransType over a longer period is expected to give a much better picture of its possibilities. We are inclined to think that after ten minutes, users were more focused on their text and less distracted by the context of the experiment and the novelty of the tool.

5.3. Behavior with respect to completions

The analysis of the completions shows that TransType imposed a certain " cognitive load" on the part of the user because of the interruption in the translation process. On average, there is 0.75s time-lag between a completion from TransType and the next action from the user. But when the user accepts a completion, the average goes up to 1.47s, almost doubling the whole average. When a completion is not accepted, a user reacts in less than 0.3s half of the time, which is quite short compared to a 0.65s average between the keystrokes measured in the first step. This can be seen clearly in figure 3, where the shapes of the two curves differ greatly. One may thus assume that users did not stop to look at the completions or did not see them.

To understand when TransType should display its completion, it is interesting to look at the average number of characters that a user types before accepting a completion. In step 2, 1535 units (71%) were entirely typed by the users, 311 (14%) were entered by a completion spanning whole word, and 100 (5%) were completed after typing only one letter. Only 10% of the words received a system-generated completion after two or more characters had been typed.

Looking at the reaction time and the length of the completions, we can conclude that when a user accepts a completion, it is at the beginning of a word most of the time. Thus the best way to improve the use of the completions would be to convince translators that they should look at the completions very soon in their typing process. As examining at these completions and deciding if they are worthwhile takes time and can in a way distract the user thinking process, completions must be valuable i.e. long enough. This is why in TT_2, we are investigating strategies where completions must exceed a certain length and level of confidence before they are proposed to the user.

5.3.1. *Length of useful completions*

As we were told by translators, it seems that completions of 3 letters or less are not very useful. We had essentially similar comments in the TT_{1a} evaluation but for TT_{1b} we can corroborate this impression with

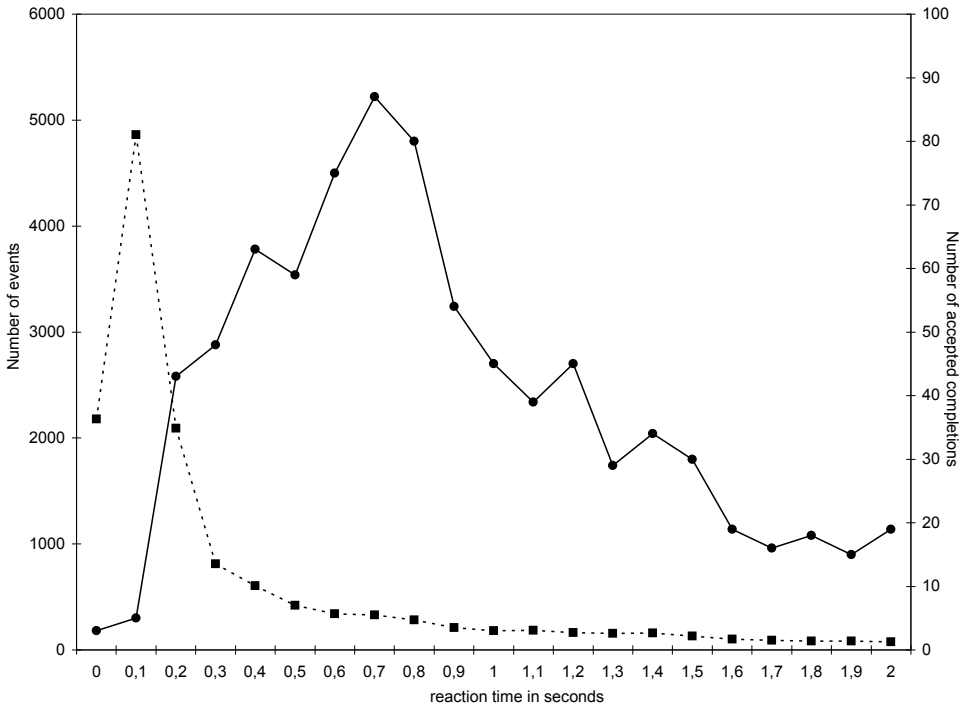

Figure 3. Reaction time with TransType. The dotted curve with the scale on the left shows the distribution of the reaction time for each event from the user. The dark curve with the scale on the right shows the distribution of reaction time when the user selects a completion from TransType.

"hard" figures by looking at the average time between two keystrokes (0.65s) and between the display of a completion and its approval (1.4s). This means that a user can type more than two characters in the time it takes to read and accept a completion. So with a completion of less than three characters, the user is bound to lose some time. We conclude that completions of less than four characters should not even be displayed.

In our experiment, the average length of accepted completions was 5.5 characters, but 42% of these were less than four characters (51% of the displayed completions had less than four characters). Given

the fact that these short completions are likely to incur some loss in productivity, we can in part explain why translators were slower with TransType (Step 2) than without (Step 1).

5.3.2. *Use of the lexicon*
In the third step of our experiment, we decided to work on a text for which we handcrafted an appropriate lexicon for units such "les diététistes du Canada" as a translation for "dieteticians of Canada".

In this step, accepted completions were longer (7.5 characters instead of 5.5) but the loss in productivity was larger than for Step 2 (see table VI). The different domain of discourse surely influenced this result. We also observed that translators often started the translation of a unit differently from the ones we had inserted in the lexicon and thus TransType could not propose an appropriate completion from its lexicon. Although the numerical results do not illustrate it clearly, interviews with the participants showed that customization with a user lexicon is very much appreciated, as we will discuss in section 5.4.

5.3.3. *Performance of the translation engine*
A great deal of effort in our project has been devoted to the development of the statistical translation engine, so it is interesting to examine its performance in a real setting. In step 2, TransType proposed an appropriate translation for 899 words (42%) that appeared in the translator's final text, and for 747 of them (35%) the first completion (the one at the top of the completion menu) was the right one. This means that TransType' s completions were "optimal" 77% of the time. Only 376 words (17%) did not receive any completions and about half of these were words with spelling errors or apostrophes; a bug in our prototype prevented TransType from properly handling apostrophes.

5.4. Qualitative evaluation

From the analysis of the answers to our post-experiment questionnaire in the two trials, it emerged that all our testers (except one) were enthusiastic about this concept of a translation typing tool, even though our prototype was far from perfect. They liked the idea that they could work at their own pace, either accepting or ignoring TransType completions, contrary to other translation tools that are always running even when they are not needed. For example, deactivating a translation memory is more cumbersome than only ignoring a completion made by TransType. The translators appreciated the fact that they did not have to check for the correct spelling of completions. Most of them were confident that with time they would become more proficient at making a better use of TransType.

The translators had more mixed feelings about the influence of TransType on the literary quality of their translations: some were under the impression that TransType induced a literal mode of translation. But they also noticed that it could have a positive effect because TransType allowed them to easily get the "long formulation" of a translation in cases where they would probably have typed an shorted form even though it would be less correct. Five out of nine said that TransType did induce some disruption in the course of their translation process; two said it was only a minor annoyance and two said that it was not a distracting factor. In fact, one translator pointed out that once he had his translation in mind, he went along without looking at the completions; and, in fact he almost never took advantage of the completions.

About half of the participants were under the impression that they were translating faster with TransType. Others mentioned it was faster to type TransType short completions than to select them. On the other hand, TransType was especially appreciated in the case when someone does not have a good idea for starting or continuing a translation.

All translators were very happy with the possibility of adding their own lexicon to TransType; for some users, this could even justify adopting it. Of course, this customization is always appreciated because users like to appropriate their tools, even though in practice very few take the time to really adapt their text editor.

6. Related Works

It is difficult to compare TransType with other MT systems because of its unique embedding of a statistical translation engine within a text editing environment.

Although the style of text prediction proposed in TransType is novel, there are numerous precedents for text prediction in a monolingual setting. Many programs such as *GNU Emacs* and *tcsh* offer built-in word or command completion features, and word-completion add-ons are also available for standard word processing environments; for example the "small floating yellow windows" that Microsoft Word pops up when a prefix of a unique known word in a special table is recognized. In this case, the strings proposed were determined either when Word was compiled or they are painstakingly added by the user. Word proposes only one possibility, while TransType determines many completions at run-time, depending on the contexts of both the target and the source texts.

Dynamic completions also occur in the field of *alternative and augmentative communication* (AAC), which deals with communication aids for the disabled, such as the *Reactive Keyboard* (Darragh and Witten, 1992) and WordQ (2001). These systems also try to guess what the user wants to type next. In this case, the completions or choices depend only on what has already been typed. In TransType, it is possible to vary the relative contributions of both the language and translation models; so in principle, we could set it up so that only the language model is used but we have not done any experiments with this option. We know however that the theoretical performance would decrease with such a scenario.

Translation memories such as the one implemented in the Translator's workbench (TranslatorsWorkbench98, 1998) also address the problem of speeding up the typing of translations. A translation memory is an interface to a database of pairs of sentences and their associated translations. Within a text editor, the translation memory manager first checks if the current sentence can be found in the source side of the database of previous translations and if so, it proposes its previous translation, which can either be accepted or modified by the translator. This environment can be quite efficient in the case of repetitive texts or for revisions of already translated texts. Although some "fuzzy matches" are allowed for finding similar but not identical sentences (for example sentences can vary by dates or numbers) this approach is not as flexible as the dynamic completions of TransType. Another drawback is the fact that once a user operates in the context of a translation memory, it is often awkward to stop it from proposing new sentences even if they are not relevant. TransType on the other side is a silent helper whose completions can be quietly ignored when the translator already knows what is to be typed.

7. Conclusion and Future Work

TransType is an innovative way of embedding machine translation; it implements an appealing interactive machine translation scenario where the interaction is mediated via the target text under production. Among other advantages, this approach relieves the translator of the burden of source text analyses, and gives him or her direct control over the final translation without having to resort to post-editing. Although some drawbacks have been identified, this user evaluation was very useful in demonstrating the interest of this innovative concept in a real setting. It is thus possible to develop computer aided translation tools

which can help to improve the efficiency of translators who are more and more in demand in the new global economy.

Although our analysis has shown that TRANSTYPE has yet to prove its efficiency, the tool is nevertheless appreciated by the users. After the first ten minutes of use, our statistics show improved performance, so we are confident that with a short learning period, translators would become much more proficient with this tool.

Our study has shown that although the translation engine performs very well at finding and displaying completions, translators do not always take advantage of them, and even, when they do, they may still lose some time. This means that much work remains to be done not only in the user-interface aspect but more importantly in the prediction engine in order to produce better completions. This is currently being addressed in TransType2 (Sema Spain et al., 2002), a follow-up project funded by the European Fifth Framework Programme in which our lab is involved with other research and user teams from Spain, Germany, France and Canada.

Acknowledgements

The TRANSTYPE project has been funded by a Strategic Grant from the Natural Sciences and Engineering Research Council of Canada in collaboration with the Machina Sapiens company. We are greatly indebted to Elliott Macklovitch, George Foster and Pierre Isabelle for the fruitful orientations they have given to this work. We also thank the referees who made many constructive suggestions for improving the organization of the paper.

References

Blanchon, H.: 1991, 'Problèmes de désambiguisation interactive et TAO personnelle'. In: *L'environnement Traductionnel*. Mons, pp. 31–48.

Brousseau, J., C. Drouin, G. Foster, P. Isabelle, R. Kuhn, Y. Normandin, and P. Plamondon: 1995, 'French speech recognition in an automatic dictation system for translators: the Transtalk project'. In: *Eurospeech 95*. Madrid, pp. 193–196.

Brown, P. F., S. A. D. Pietra, V. D. J. Pietra, and R. L. Mercer: 1993, 'The Mathematics of Machine Translation: Parameter Estimation'. *Computational Linguistics* **19**(2), 263–312.

Brown, R. D. and S. Nirenburg: 1990, 'Human-Computer Interaction for Semantic Disambiguation'. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Helsinki, Finland, pp. 42–47.

Chandioux, J.: 1989, 'Météo: 100 million words later'. In: D. H. (ed.) (ed.): *American Translators Association Conference: Coming of Age, Learned Information*. Medford NJ:, pp. 449–453.

Darragh, J. J. and I. H. Witten: 1992, *The Reactive Keyboard*. Cambridge University Press.

Foster, G.: 2000, 'Incorporating Position Information into a Maximum Entropy/Minimum Divergence Translation Model'. In: *4th Computational Natural Language Learning Workshop*. Lisbon, Portugal.

Foster, G., P. Isabelle, and P. Plamondon: 1997, 'Target-Text Mediated Interactive Machine Translation'. *Machine Translation* **12**, 175–194.

Jelinek, F.: 1990, 'Self-organized Language Modeling for Speech Recognition'. In: A. Waibel and K. Lee (eds.): *Readings in Speech Recognition*. San Mateo, California: Morgan Kaufmann, pp. 450–506.

Kay, M.: 1973, 'The MIND System'. In: R. Rustin (ed.): *Natural Language Processing*. New York: Algorithmics Press, pp. 155–188.

Kay, M.: 1996, 'Machine Translation: The Disappointing Past and Present'. In: R. A. Cole (ed.): *Survey of the State of the Art in Human Language Technology*. `http://cslu.cse.ogi.edu/HLTsurvey/ch8node4.html`: Center for Spoken Language Understanding, Oregon Graduate Institute, USA, Chapt. 8.2.

Langlais, P. and G. Foster: 2000, 'Using Context-Dependent Interpolation to Combine Statistical Language and Translation Models for Interactive Machine Translation'. In: *Computer-Assisted Information Retrieval*. Paris, pp. 507–518.

Langlais, P., G. Foster, and G. Lapalme: 2000a, 'Unit Completion for a Computer-aided Translation Typing System'. *Machine Translation* **15**, 267–294.

Langlais, P., G. Foster, and G. Lapalme: 2001a, 'Integrating Bilingual Lexicons in a Probabilistic Translation Assistant'. In: *MT Summit VIII,*. Santiago de Compostela, Spain, pp. 197–202.

Langlais, P., G. Lapalme, and S. Sauvé: 2001b, 'User Interface Aspects of a Translation Typing System'. In: E. Stroulia and S. Matwin (eds.): *AI2001 - Advances in Artificial Intelligence*. Ottawa, Ontario, pp. 246–256.

Langlais, P., S. Sauvé, G. Foster, E. Macklovitch, and G. Lapalme: 2000b, 'Evaluation of TransType, a Computer-aided Translation Typing System: A comparison of theoretical- and user- oriented evaluation procedures'. In: *Second International Conference On Language Resources and Evaluation (LREC)*, Vol. 2. Athens, Greece, pp. 641–648.

Maruyama, H. and H. Watanabe: 1990, 'An Interactive Japanese Parser for Machine Translation'. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Helsinki, Finland, pp. 257–262.

Miller, G. A.: 1956, 'The magical number seven, plus or minus two: Some limits on our capacity for processing information'. *Psychological Review* **63**, 81–97.

Sema Spain, Computer Science Department RWTH Aachen - University of Technology, Instituto Tecnológico de Informática, Universidad Politécnica de Valencia, RALI Laboratory - University of Montreal, Celer Soluciones, Société Gamma, and Xerox Research Centre Europe: 2002, 'Transtype2 - Computer Assisted Translation Project funded by the European Commission under the IST Programme (IST-2001-32091)'. `http://tt2.sema.es/`.

TranslatorsWorkbench98: 1998, 'Trados Translator's Workbench 2, Product Description'. `www.trados.com/workbench/index.html`.

Whitelock, P. J., M. M. Wood, B. J. Chandler, N. Holden, and H. J. Horsfall: 1986, 'Strategies for Interactive Machine Translation: the experience and implications of the UMIST Japanese project'. In: *Proceedings of the International Conference on Computational Linguistics (COLING)*. Bonn, West Germany, pp. 329–334.

WordQ: 2001, 'Product Description'. `www.wordq.com`.